Indice

Introduzione

La radioterapia nasce ufficialmente nel 1922, oltre 25 anni dopo le scoperte dei raggi X (Roentgen, 1895) e del radium (Curie, 1896), a Parigi, dove, al Congresso Internazionale di Oncologia, furono presentate le prime evidenze della sua efficacia nel trattamento di differenti neoplasie. Da allora, e specialmente negli ultimi venti anni, si è assistito ad una crescita esponenziale delle conoscenze e delle applicazioni della radioterapia. Questo sviluppo è stato dovuto sia ai miglioramenti metodologici nell'affrontare la terapia oncologica, con l'integrazione tra le differenti discipline e le scienze di base, sia allo sviluppo tecnologico che ha consentito di ottenere un elevato livello di qualità delle tecniche impiegate.
Lo scopo della radioterapia è depositare, attraverso radiazioni o fasci di particelle, una quantità di energia nella massa tumorale tale da uccidere le cellule malate, ma con una precisione spaziale da preservare quelle sane, così da non danneggiare in modo grave il paziente. Importante risulta dunque la necessità di perfezionare i piani di trattamento, attraverso la ricerca di tecniche, basate sulla simulazione virtuale, mirate a calcolare le distribuzioni ottimali di dose al bersaglio.
I continui miglioramenti in questo ambito hanno portato alla nascita di molti centri e istituti di ricerca, tra i quali si colloca la Fondazione TERA, oggi formata da più di centocinquanta studiosi, fra medici, fisici, ingegneri e informatici, provenienti da quaranta istituti differenti (università, ospedali, centri oncologici e laboratori di ricerca italiani), nata con lo scopo di portare per l'anno 2005 l'Italia all'avanguardia nel campo della terapia dei tumori con le radiazioni, attraverso la costruzione a Milano del primo Centro di Adroterapia Italiano.
A questa fondazione va anche il merito della progettazione e creazione di camere a ionizzazione utilizzate in dosimetria, tra cui la Camera Monitor a Pixel, ancora in via di sviluppo, realizzata per ottenere in tempo reale una ricostruzione geometrica precisa in due dimensioni del fascio.
In tale ambito si inserisce questo lavoro di tesi, finalizzato allo studio e allo sviluppo di un codice di simulazione della camera per ottimizzare i parametri di costruzione e ottenere risultati in accordo con i dati sperimentali. Il simulatore realizzato utilizza il toolkit fornito da Geant4 ed è stato implementato utilizzando il linguaggio di programmazione C++.

Più in particolare in questa tesi verrà focalizzata l'attenzione sui seguenti argomenti:

- **le interazioni** della radiazione con la materia, approfondendo gli effetti fisici riguardanti i fotoni, gli elettroni e gli adroni, il calcolo dell'energia depositata e delle sezioni d'urto utilizzate dal codice di simulazione.
- **la radioterapia**, le sue basi fisiche e le grandezze fondamentali, i fasci e gli acceleratori utilizzati nei piani di trattamento, l'analisi delle terapie attuali e degli eventuali sviluppi futuri.
- **Geant4**, l'analisi e la descrizione della sua struttura e delle principali classi necessarie allo sviluppo di un codice di simulazione.
- **la Camera Monitor a Pixel**, la sua struttura e le sue funzionalità, il codice implementato, l'analisi e la risoluzione del problema della penombra, presente nel profilo del fascio.
- **i risultati**, ottenuti dalle simulazioni della camera in diverse configurazioni, confrontati con i dati sperimentali. Le principali misure ottenute riguardano il calcolo dell'errore statistico, della penombra del fascio di fotoni e della curva di build-up, i profili ricavati dal gantry rotante e il calcolo dell'output factor, riguardo al quale è stata implementata la simulazione della camera Farmer.
- **misura del picco di Bragg** per protoni incidenti in acqua e in una lastra di monomeri, ottenuto dalla simulazione del fantoccio ad acqua. Questo studio è stato effettuato in collaborazione con l'Istituto di Fisica di Roma.

Come verrà descritto in seguito, il codice di simulazione viene configurato attraverso un insieme di parametri che, basandosi esclusivamente sul toolkit Geant4, devono essere modificati agendo direttamente sul codice o su un file macro. E' evidente dunque che eseguire il simulatore su una molteplicità di test può risultare laborioso. Al fine di ridurre ed automatizzare il processo di configurazione è anche stato progettato e sviluppato un modulo di interfaccia grafica che permette di impostare i valori dei parametri, eseguire il simulatore e visualizzarne i risultati. Questo codice è stato implementato utilizzando il linguaggio di programmazione JAVA per sfruttare il vantaggio della portabilità su piattaforme diverse e rendere trasparente all'utente l'implementazione del simulatore.

Capitolo 1

Interazioni elettromagnetiche

In questo capitolo viene presentata la teoria delle interazioni elettromagnetiche fra particella e materia, analizzando gli effetti fisici prodotti da fotoni, elettroni e adroni incidenti. Sono state considerate le sezioni d'urto dei vari processi che permettono al codice di simulazione utilizzato, Geant4, di calcolare il deposito di energia nella materia.

Introduzione

Il passaggio di particelle nella materia provoca rilascio di energia a seguito di interazioni determinate dal tipo e dall'energia della particella incidente: vengono definite *radiazioni direttamente ionizzanti* i fasci di particelle cariche che cedono energia attraverso processi elettromagnetici, *radiazioni indirettamente ionizzanti* se si tratta di particelle neutre e fotoni che cedono energia a particelle secondarie direttamente ionizzanti.
Le interazioni si distinguono in *elettromagnetiche*, quando intervengono campi coulombiani, e *adroniche*, invece, se è in gioco la forza nucleare forte. In questo lavoro di tesi sono stati utilizzati fasci di fotoni, quindi sono state prese in considerazione solo le prime. La perdita di energia per interazioni elettromagnetiche è predominante fino a circa 100 MeV, poi per gli adroni carichi diventa considerevole il contributo delle interazioni nucleari.
Le particelle cariche interagiscono con processi elettromagnetici tramite collisioni coulombiane inelastiche e elastiche con gli atomi del mezzo attraversato. Le prime comportano una continua perdita di energia da parte della particella incidente che va ad eccitare atomi e molecole del materiale; questi si diseccitano emettendo

fotoni, elettroni Auger, ionizzando il mezzo o attraverso moti vibrazionali e rotazionali. Le collisioni elastiche invece causano la diffusione laterale della particella incidente (*multiple scattering*) senza una consistente perdita di energia, l'effetto è più evidente quanto più è piccola la massa della particella incidente. Per masse maggiori o uguali a quella dell'elettrone diventano più importanti altri fenomeni, come la radiazione di frenamento (*bremsstrahlung*), l'effetto Čerenkov o la radiazione di transizione. Saranno dunque trattati in modo differente fotoni, elettroni e particelle più pesanti, come protoni, ioni, muoni, ...

1.1 Fotoni

Nell'attraversare uno spessore x di materiale, un fascio di fotoni di intensità I_0 viene assorbito secondo la legge esponenziale:

$$I(x) = I_0 \; e^{-\mu \; x} = I_0 \; e^{-\frac{\mu}{\rho} \; X} \tag{1.1}$$

con μ *coefficiente di attenuazione lineare* definito dalla relazione $\mu = N\sigma$, dove N è il numero di atomi per unità di volume e σ è la sezione d'urto totale. Il passaggio di raggi γ nella materia provoca tre principali fenomeni:

1. effetto fotoelettrico
2. effetto Compton
3. creazione di coppie

1.1.1 Effetto fotoelettrico

Nell'assorbimento fotoelettrico l'energia del fotone, E_γ, viene spesa per liberare un elettrone di un livello interno dell'atomo assorbitore. Se l'energia di legame dell'elettrone è $B_{shell}(Z)$, allora l'energia T_{phe} dell'elettrone liberato è data da:

$$T_{phe} = E_\gamma - B_{shell}(Z)$$

Naturalmente il processo avviene solo se il fotone ha un'energia E_γ maggiore di quella di legame $B_{shell}(Z)$ dell'elettrone interessato. Il posto vuoto creatosi viene così occupato da un elettrone esterno che si avvicina al nucleo, emettendo un fotone di energia pari al salto energetico fra le shell. Se il γ viene emesso si ha

il fenomeno della *fluorescenza*, altrimenti può essere assorbito da un elettrone di un orbitale esterno che esce dall'atomo (*elettroni Auger*).
L'effetto fotoelettrico prevale per Z elevati e per energie $E_\gamma < 100\ keV$ ed è associato ad una sezione d'urto atomica[1]

$$\sigma_f(Z, \alpha) \propto \alpha^4 r_0^2\ Z^5 \tag{1.2}$$

dove

- $\alpha = \frac{1}{137}$ è detta *costante di struttura fine*
- $r_0 = \frac{e^2}{m\ c^2} = 2.82$ fm è detto *raggio classico dell'elettrone*

Se si considera un materiale monoatomico, si può indicare il numero di atomi per unità di volume come

$$n = \frac{N_A\ \rho}{A}$$

con:

- N_A numero di Avogadro
- ρ densità del mezzo
- A massa atomica

Se si considera invece un materiale poliatomico, si può indicare il numero di atomi di un elemento *elm* come

$$n_{elm} = \frac{N_A\ \rho\ w_{elm}}{A_{elm}}$$

con:

- w_{elm} frazione in massa dell'elemento *elm*
- A_{elm} massa atomica dell'elemento *elm*

Si definisce allora *libero cammino medio* di un fotone che interagisce con effetto fotoelettrico[2], cioè lo spazio medio percorso da un γ prima che venga assorbito, come

$$\lambda(E_\gamma) = \frac{1}{n\ \sigma(Z, E_\gamma)} = \frac{1}{\sum_{elm}\ [n_{elm}\ \sigma(Z_{elm}, E_\gamma)]} \tag{1.3}$$

dove viene effettuata la somma su tutti gli elementi del materiale assorbitore.

Figura 1.1: Schema del processo fotoelettrico in un atomo.

1.1.2 Effetto Compton

L'effetto Compton avviene quando il fotone incidente viene diffuso da un elettrone libero o appartenente ad un orbitale esterno e cede ad esso solo parte della sua energia in una collisione elastica. Il fotone viene deflesso di un angolo θ e l'energia E'_γ è pari a

$$E'_\gamma = \frac{E_\gamma}{1 + (1 - cos\theta)\ \epsilon}$$

con $\epsilon = \frac{E_\gamma}{mc^2}$.
L'elettrone di rinculo avrà un'energia pari a

$$E'_{e^-} = \epsilon\ E_\gamma\ \frac{1 - cos\theta}{1 + (1 - cos\theta)\ \epsilon}$$

L'effetto Compton prevale per energie del fotone incidente comprese fra 0,5 e 10 MeV ed è associato ad una sezione d'urto che è descritta dalla formula di Klein e Nishina[2]

$$\sigma_c(E_\gamma, E'_\gamma) = \frac{X_0\ n\ \pi\ r_0^2\ m_e c^2}{E_\gamma^2}[\frac{1}{\varepsilon} + \varepsilon]\ [1 - \frac{\varepsilon\ sin^2\theta}{1 + \varepsilon^2}] \tag{1.4}$$

con

- E_γ = energia del fotone incidente

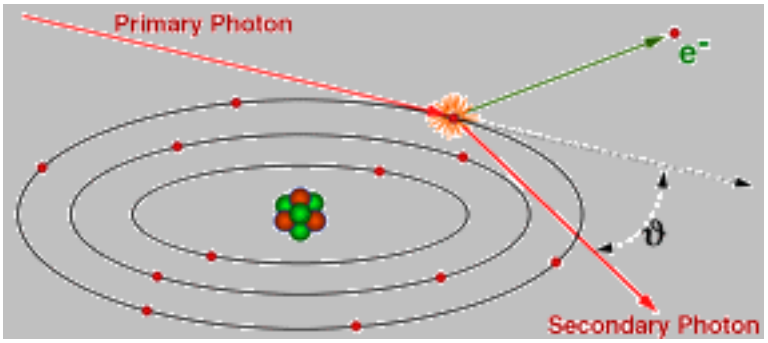


Figura 1.2: Schema dell'effetto Compton in un atomo.

- E'_γ = energia del fotone diffuso
- $\varepsilon = \frac{E_\gamma}{E'_\gamma}$ rapporto fra le energie prima e dopo l'effetto Compton
- n = numero di elettroni per unità di volume
- X_0 = lunghezza di radiazione data dalla relazione $\frac{1}{X_0} = 4\ Z^2 n \alpha r_0^2\ log(\frac{183}{Z^{\frac{1}{3}}})$

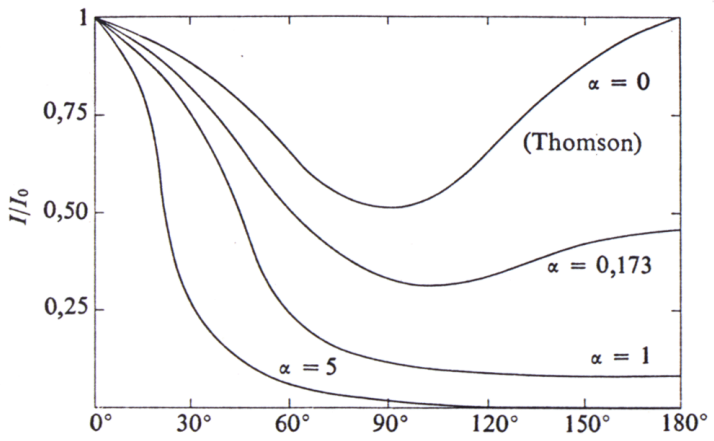


Figura 1.3: Distribuzione angolare dei raggi X diffusi per effetto Compton. La variabile α indica il rapporto $\alpha = \hbar\omega/2\pi mc^2$.

1.1.3 Creazione di coppie

La trasformazione di un fotone in $e^- + e^+$ è anche detta *materializzazione*. Perchè questo fenomeno avvenga è necessaria la presenza di un nucleo o di un elettrone che generi un campo elettromagnetico, solo così un γ di energia maggiore all' energia di soglia $E_s = 2m_e c^2 = 1,02$ MeV può dare origine ad una coppia.

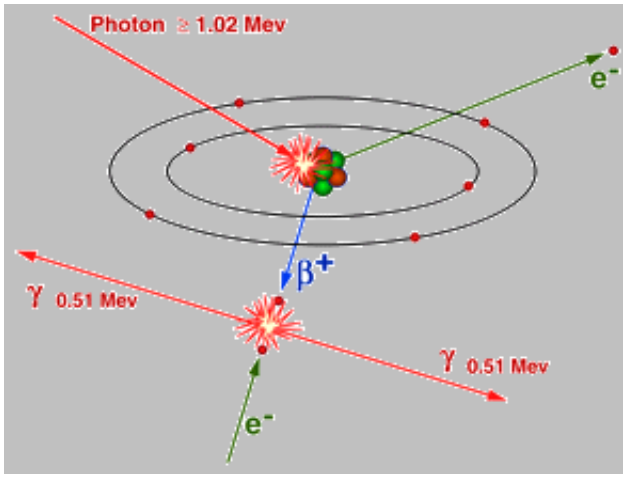


Figura 1.4: Schema del processo di creazione di coppie in un atomo.

L'energia del fotone incidente è trasferita nella massa di riposo delle due particelle prodotte, nella loro energia cinetica e nell'energia di rinculo del nucleo che genera il campo:

$$E_\gamma = h\nu = 2m_e c^2 + T_{e^-} + T_{e^+} + E_R$$

Il positrone creato successivamente può annichilarsi con un elettrone atomico e generare due γ di energia uguale, pari a $E'_\gamma = m_e c^2$. I fotoni hanno direzioni opposte per la conservazione dell'impulso.
La creazione di coppie ha una sezione d'urto, per energie relativistiche, proporzionale a $r_0^2 Z^2$. Nel caso di energie alte, $E_\gamma = h\nu \gg m_e c^2$, è stata calcolata essere[1]

$$\sigma_p = \frac{e^2}{\hbar c}\, Z^2\, r_0^2 \left(\frac{28}{9}\, log\, \frac{183}{Z^{1/3}} - \frac{2}{27} \right) \tag{1.5}$$

Il libero cammino medio del fotone per generazione di coppie è

$$X_{cp} = \frac{9}{7} X_0$$

con X_0 lunghezza di radiazione.

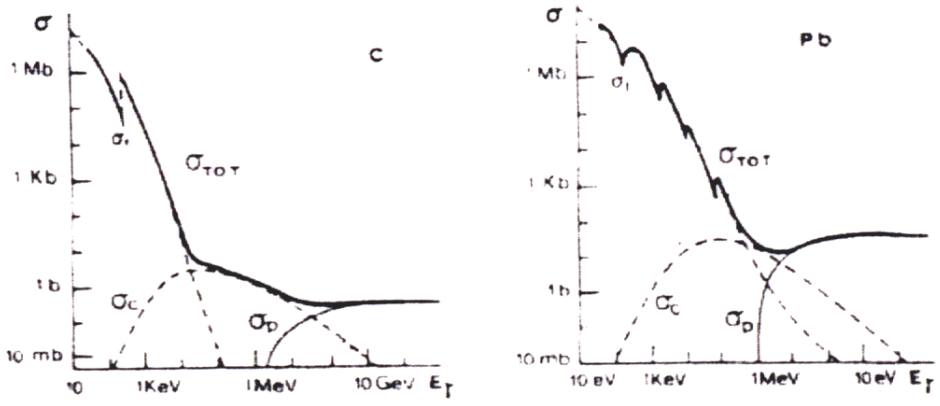


Figura 1.5: Sezione d'urto totale, ricavata dalle sezioni d'urto dell'effetto fotoelettrico, dell'effetto Compton e della creazione di coppie, per fotoni rispettivamente su C e Pb.

1.1.4 Effetti secondari

Meno importanti sono invece altri due fenomeni che interessano il passaggio di raggi γ nella materia:

- **la diffusione Rayleigh** prodotta sui fotoni da particelle con dimensioni piccole rispetto alla lunghezza d'onda λ della luce. L'intensità della radiazione diffusa è descritta dalla relazione $I \propto \frac{1+cos^2\theta}{\lambda^4}$ (quindi sarà diffusa 16 volte tanto una luce blu a cui corrisponde una $\lambda \approx 400$ nm rispetto ad una luce rossa che ha una $\lambda \approx 800$ nm).

- **l'effetto fotonucleare**, anche chiamato *fotodisintegrazione* o *fotoproduzione*, dato dall'azione di fotoni ad elevata energia che colpiscono un nucleo liberando un protone, un neutrone o altre particelle.

1.2 Elettroni

La perdita di energia degli elettroni è determinata da due contributi fondamentali: la ionizzazione del materiale attraversato e la radiazione elettromagnetica (*radiazione di bremsstrahlung*), emessa nelle violente accelerazioni che si verificano durante gli urti.

1.2.1 Ionizzazione

L'elettrone viaggiando nella materia può urtare contro altri e^- atomici che vengono strappati dal loro legame, l'atomo diventa così uno ione positivo fino a quando non catturerà un altro elettrone.

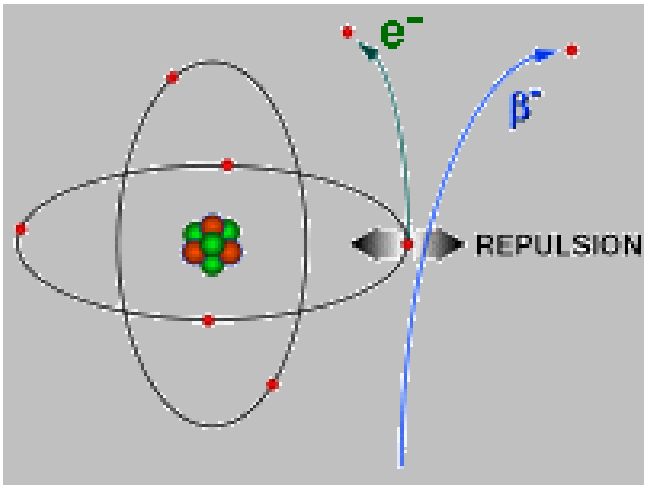


Figura 1.6: Schema del processo di ionizzazione.

La perdita di energia dovuta alla ionizzazione del mezzo dopo un percorso dx è data dalle seguenti relazioni[2]:

- per elettroni non relativistici

$$-\frac{1}{\rho}\left(\frac{dE}{dx}\right)_{ion} = \frac{2\pi e^4 N_A}{m_e c^2 \beta^2}\left[ln\left(\frac{m_e v^2}{2\sqrt{2}\ I}\right) + \frac{1}{2}\right] \tag{1.6}$$

- per elettroni relativistici

$$-\frac{1}{\rho}\left(\frac{dE}{dx}\right)_{ion} = \frac{2\pi e^4 N_A}{m_e c^2 \beta^2}\left[ln\left(\frac{2m_e\gamma^{3/2}v^2}{I}\right) - \frac{1}{2}ln8 + \frac{1}{16} - \frac{C}{Z} - \frac{\delta}{2}\right] \quad (1.7)$$

dove si definisce con $\frac{1}{\rho}\left(\frac{dE}{dx}\right)$ *il potere frenante di massa*, che decresce con l'aumentare di Z ed è funzione di n_e/ρ e di I. A volte è anche usato $\left(\frac{dE}{dx}\right)$ detto *potere frenante lineare.*

Le traiettorie degli elettroni nella materia non sono però rettilinee come quelle di particelle più pesanti, soprattutto a basse energie ($E \ll m_e c^2$), quindi il cammino effettivo di un e^- tra due punti può essere maggiore in modo considerevole rispetto alla distanza fra due i punti.

Elettroni della stessa energia allora non si fermeranno tutti dopo uno stesso spessore, dunque il concetto di *percorso* perde di significato. Per misurare la perdita di energia viene utilizzato il *percorso estrapolato*, o *pratical range*, stimabile dalla seguente relazione semiempirica:

$$R_p = 0,71\ E^{1,71}\ \ g\ cm^{-2} \quad (1.8)$$

La sezione d'urto per ionizzazione viene calcolata dall'integrazione della seguente formula:

$$\sigma(E, T_{cut}) = \int_{T_{cut}}^{T_{max}} \frac{d\sigma(E,T)}{dT}\, dT \quad (1.9)$$

dove $\frac{d\sigma}{dT}$ è la sezione d'urto differenziale per l'emissione di un elettrone di energia cinetica T da parte di un elettrone incidente con energia E che attraversa un materiale di densità ρ. T_{cut} è il valore di taglio dell'energia cinetica, sotto la quale la perdita di energia viene considerata continua, mentre T_{max} è la massima energia trasferibile ad un elettrone libero e vale $(E - m_e)$ per gli e^-, $\frac{(E-m_e)}{2}$ per i e^+. Otteniamo così le sezioni d'urto[2]:

- per lo scattering Möller (e^-, e^-) con energia di soglia $T_s^{Mol} = 2T_{cut}$

$$\sigma(Z, E, T_{cut}) = \frac{2\pi r_0^2 m_e Z}{(E - m_e)\beta^2}\left[\frac{(\gamma-1)}{\gamma^2}\left(\frac{1}{x} - 1\right) + \frac{1}{x} - \frac{1}{1-x} - \frac{2\gamma - 1}{\gamma^2} ln\frac{1-x}{x}\right]$$

- per lo scattering Bhabha (e^+, e^-) con energia di soglia $T_s^{Bh} = T_{cut}$

$$\sigma(Z, E, T_{cut}) = \frac{2\pi r_0^2 m_e Z}{(E - m_e)} \left[\frac{1}{\beta^2} \left(\frac{1}{x} - 1 \right) + B_1 x + B_2(1 - x) - \frac{B_3}{2}(1 - x^2) + \frac{B_4}{3}(1 - x^3) \right]$$

con

$\gamma = \frac{E}{m_e}$ $\qquad$ $B_1 = 2 - y^2$

$\beta^2 = 1 - 1/\gamma^2$ $\qquad$ $B_2 = (1 - 2y)(3 + y^2)$

$x = \frac{T_{cut}}{E - m_e}$ $\qquad$ $B_3 = (1 - 2y)^2 + (1 - 2y)^3$

$y = \frac{1}{\gamma + 1}$ $\qquad$ $B_4 = (1 - 2y)^3$

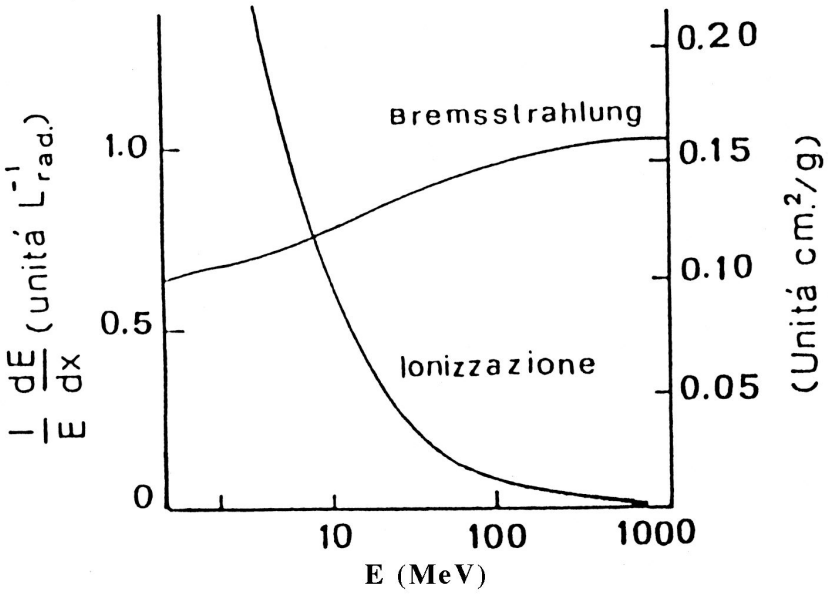


Figura 1.7: Perdita percentuale di energia per unità di percorso di elettroni in Pb.

1.2.2 Radiazione di bremsstrahlung

Questo fenomeno diventa importante per elettroni veloci ($E \gg m_e c^2$) che, attraversando la materia, emettono radiazioni elettromagnetiche a causa delle accelerazioni a cui sono soggetti. La perdita di energia per radiazione è proporzionale all'energia dell'elettrone E_e e a Z^2, invece la perdita di energia per ionizzazione aumenta linearmente con Z e solo logaritmicamente con E_e.
Il rapporto tra la perdita di energia per bremsstrahlung e quella dovuta alla ionizzazione è descritto dalla seguente relazione:

$$\frac{\left(\frac{dE}{dx}\right)_{brem}}{\left(\frac{dE}{dx}\right)_{ion}} = \frac{Z\ E_e}{580}$$

e cresce con l'aumentare dell'energia degli elettroni incidenti. Viene definita un'energia critica E_c in corrispondenza della quale le due perdite di energia si equivalgono:

$$E_c = \frac{580}{Z} \quad MeV$$

La perdita di energia per bremsstrahlung è stata calcolata da Fermi per un e^- di velocità $v \simeq c$ che viaggia lungo z e passa in prossimità di un nucleo di carica Ze. Nel sistema di riferimento in quiete, sull'elettrone, si vedrà un campo elettrico **E** perpendicolare alla direzione del moto e un campo magnetico **H** perpendicolare ad **E** e a z[1]. Questi due campi formano un'onda piana che viene diffusa sull'elettrone, e i quanti diffusi appaiono nel sistema del laboratorio come radiazione di bremsstrahlung. La perdita media di energia calcolata per un e^- che attraversa una sostanza i cui atomi hanno numero atomico Z è

$$-\left\langle \frac{dE}{dx} \right\rangle_{brem} = \frac{4\alpha\ Z(Z+1)\ N_A\ r_0^2}{A}\ E\ ln\ \frac{183}{Z^{1/3}} = \frac{E}{X_0} \qquad (1.10)$$

dove X_0 è la *lunghezza di radiazione* dell'assorbitore, cioè lo spessore di materia necessario per ridurre l'energia dell'elettrone di un fattore e.
Infatti integrando l'eq. (1.10) si ottiene l'espressione per l'energia:

$$E = E_0\ e^{-\frac{x}{X_0}}$$

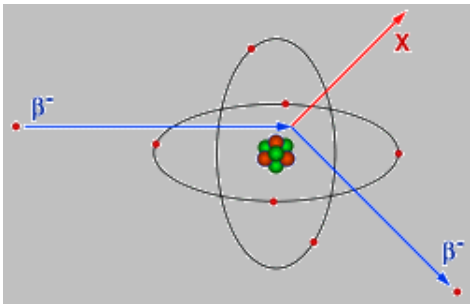


Figura 1.8: Schema dell'effetto di Bremsstralung in un atomo.

La sezione d'urto, per la produzione di un γ di energia k da parte di un elettrone di energia cinetica T, in presenza di un campo elettromagnetico generato da un atomo di carica Z, è data dall'integrale

$$\sigma_{brem}(Z,T,k_c) = \int_{k_c}^{T} \frac{d\sigma(Z,T,k)}{dk}\, dk \tag{1.11}$$

dove k_c è l'energia di taglio sotto la quale la perdita di energia del fotone è considerata continua. Per il calcolo delle sezioni d'urto $\sigma(Z,T,k)$ vengono utilizzati i tabulati di Seltzer e Berger, che riportano $\frac{d\sigma}{dk}$ per l'emissione di fotoni di energia k da parte di elettroni di energia cinetica T compresa fra 1 keV e 10 GeV.

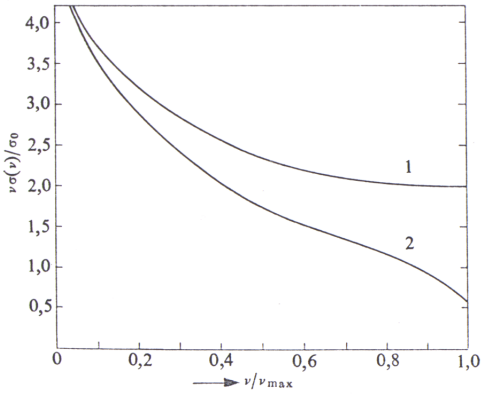


Figura 1.9: Sezione d'urto integrale di Bremsstralung in funzione della frequenza normalizzata rispetto al valore massimo. La curva 1 si riferisce ad elettroni di $E = 27\ Z^2\ eV$, la curva 2 a $E = 4300\ Z^2\ eV$.

Sciami elettromagnetici

Strettamente connesso alla bremsstrahlung è il fenomeno della produzione di coppie, quando i γ emessi sono abbastanza energetici per generare un elettrone e un positrone. Radiazione elettromagnetica e creazione di coppie insieme si combinano per generare il fenomeno degli **sciami**: un e^- di alta energia dà inizio al processo moltiplicativo emettendo fotoni per irragiamento, questi generano coppie e^-/e^+ abbastanza energetiche perchè a loro volta emettano dei γ e il processo si ripete. La conservazione della quantità di moto determina un asse dello sciame che coincide con la direzione della particella iniziale: lo sciame si allarga lateralmente molto meno di quanto si propaga longitudinalmente.
Un elettrone di energia E dà origine in una lunghezza di radiazione a circa tre fotoni che, a loro volta, in un libero cammino medio, producono circa tre coppie. Il numero di particelle N dello sciame cresce quindi esponenzialmente secondo la legge:

$$N(x) = N_0 \; e^{\gamma x}$$

con γ tale che, dati X_0 lunghezza di radiazione e X_{cp} libero cammino medio dei fotoni per la produzione di coppie,

$$se \;\; x = X_0 + X_{cp} = X_0 + \frac{9}{7}X_0 = \frac{16}{7}X_0 \qquad \Rightarrow \qquad N = 6$$

Si ottiene allora

$$\gamma = \frac{0,78}{X_0}$$

Il numero di particelle nello sciame aumenta fino a che l'energia non comincia a dissiparsi per ionizzazione e effetto Compton, gli elettroni hanno ormai raggiunto l'energia critica E_c e lo sciame il massimo numero di particelle, ad una distanza dall'origine pari a:

$$X_c = \frac{X_0}{0,78} \; log \; \frac{E_0}{6 \; E_c}$$

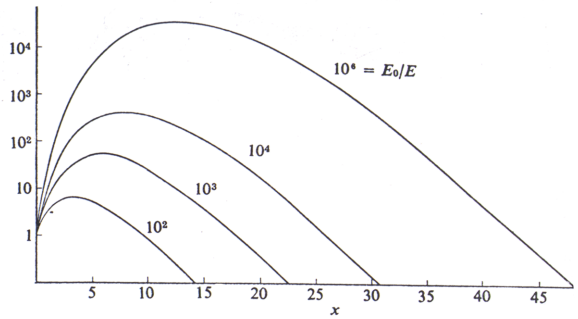


Figura 1.10: Numero medio di elettroni di energia maggiore di E di uno sciame generato da un elettrone di energia E_0 in funzione della profondità x in lunghezze di radiazione.

Il numero di particelle massimo nello sciame è

$$N_{max} = \frac{E_0}{6\ E_c} = e^{0,78\ \frac{X_c}{X_0}}$$

Queste espressioni vengono corrette per alte energie e bassi Z, otteniamo così le relazioni

$$X_c = 1,01\ X_0\ \left(\ log\ \frac{E_0}{E_c} - 1\ \right)$$

$$N_{max} = 0,31\ \frac{E_0}{E_c}\ \left(\ log\ \frac{E_0}{E_c} - 0,37\ \right)^{-1/2}$$

Esiste un modello che riproduce lo sciame elettromagnetico con il metodo MonteCarlo in cui $N \propto e^x$: il metodo dà buoni risultati per spessori piccoli durante lo sviluppo dello sciame. Non riproduce bene i dati sperimentali invece nell'intervallo di spessori in cui lo sciame si spegne, perchè la variazione del numero delle particelle in funzione dello spessore termina bruscamente ad un valore $X_{modMC} < X_{sper} \simeq 20\ X_0$.

1.2.3 Raggi δ

Vengono definiti raggi δ gli elettroni secondari prodotti per ionizzazione che hanno energia sufficiente per ionizzare a loro volta. Posto I il potere di ionizzazione del mezzo, vengono quindi detti raggi δ tutti gli elettroni con energia compresa fra I e $T_{max} = 2m_e\gamma^2\beta^2c^2$.
La distribuzione di elettroni δ è data dalla relazione:

$$\frac{d^2\ N}{dTdx} = \frac{2\pi N_A r_0^2 m_e c^2 z^2}{\beta^2}\frac{Z}{A}\frac{F(T)}{T^2} \tag{1.12}$$

dove F(T) è una funzione dipendente dallo spin dell'elettrone

- spin $0 \Longrightarrow F(T) = 1 - \beta^2\ \frac{T}{T_{cut}}$
- spin $\frac{1}{2} \Longrightarrow F(T) = 1 - \beta^2\ \frac{T}{T_{cut}} + \frac{1}{2}\left(\ \frac{T}{E+mc^2}\right)^2$
- spin $1 \Longrightarrow F(T) = \left(\ 1 - \beta^2\ \frac{T}{T_{cut}}\ \right)\left(\ 1 + \frac{T}{3E_c}\ \right) + \frac{1}{3}\left(\ \frac{T}{E+mc^2}\ \right)^2\left(\ 1 + \frac{T}{2E_c}\ \right)$

dove E e m sono l'energia e la massa della particella ionizzante, $E_c = \frac{m^2c^2}{m_e}$ è definita come energia critica e T_{cut} è il valore di soglia sotto il quale non si ha più produzione di raggi δ. Questi perdono energia per ionizzazione, infatti non sono mai abbastanza energetici per produrre radiazione di bremsstrahlung ($E_\delta \ll \frac{ZE}{580}$).
La perdita di energia per i raggi δ è data dalla seguente relazione:

$$-\left.\frac{dE}{dx}\right|_{T<T_{cut}} = \frac{4\pi m_e c^2 r_0^2 N_A z^2}{\beta^2}\ \frac{Z}{A}\ \rho\ L(\beta) \tag{1.13}$$

con

$$L(\beta) = \frac{1}{2}\ ln\ \left(\ \frac{2m_e\gamma^2\beta^2c^2T_{min}}{I^2}\ \right) - \frac{\beta^2}{2}\ \left(\ 1 + \frac{T_{min}}{T_{max}}\ \right) - \frac{\delta}{2} - \frac{C}{Z}$$

dove $T_{min} = min(T_{cut}, T_{max})$. La (1.13) è detta *formula di Bethe-Bloch ristretta.*
La sezione d'urto della produzione di raggi δ si differenzia a seconda che l'elettrone δ derivi dallo scattering fra e^-/e^+ (*Bhabha scattering*)

$$\frac{d\sigma}{d\varepsilon} = \frac{2\pi m_e Z r_0^2}{(E - m_e)}\ \left[\ \frac{1}{\beta^2\varepsilon^2} - \frac{B_1}{\varepsilon} + B_2 - B_3\varepsilon + B_4\varepsilon^2\ \right] \tag{1.14}$$

o dallo scattering fra e^-/e^- (*Möller scattering*)

$$\frac{d\sigma}{d\varepsilon} = \frac{2\pi m_e Z r_0^2}{(E - m_e)} \left[\frac{(\gamma - 1)^2}{\gamma^2} + \frac{1}{\varepsilon} \left(\frac{1}{\varepsilon} - \frac{2\gamma - 1}{\gamma^2} \right) + \frac{1}{1 - \varepsilon} \left(\frac{2\gamma - 1}{\gamma^2 (1 - \varepsilon)} \right) \right] \quad (1.15)$$

con

$\gamma = \frac{E}{M}$ $\qquad$ $B_1 = 2 - y^2$

$\beta^2 = 1 - 1/\gamma^2$ $\qquad$ $B_2 = (1 - 2y)(3 + y^2)$

$\varepsilon = \frac{T}{E - m_e}$ $\qquad$ $B_3 = (1 - 2y)^2 + (1 - 2y)^3$

$y = \frac{1}{\gamma + 1}$ $\qquad$ $B_4 = (1 - 2y)^3$

dove Z è il numero atomico del mezzo, M e E rispettivamente la massa e l'energia della particella incidente e T l'energia cinetica dell'elettrone scatterato. ε può variare fra $\varepsilon_0 = \frac{T_{cut}}{E - m_e}$ e $\frac{1}{2}$ per lo scattering Möller, fino ad 1 per la formula di Bhabha.

1.2.4 Annichilazione

Il processo di annichilazione fra un e^- e un e^+ determina la scomparsa delle due particelle e la formazione di due fotoni ciascuno di energia $E_\gamma = h\nu = m_e c^2$ che si propagano in direzioni opposte per la conservazione dell'impulso. In realtà si possono verificare anche processi più complicati, ad esempio l'annichilazione in tre gamma, che in questa tesi non verranno trattati.

Figura 1.11: Schema del processo di annichilazione di e^- e e^+ in due γ.

Considerando un e^+ incidente con energia cinetica T, energia totale $E_{e^+} = T + m_e c^2$ e momento $pc = [T(T + 2m_e c^2)]^{1/2}$ che, interagendo con un e^- di energia $E_{e^-} = m_e c^2$, origina due fotoni γ_a e γ_b di energia E_a e E_b e momento p_a e p_b, si avrà

$$E_{tot} = E_{e^+} + E_{e^-} = T + 2m_e c^2 = E_a + E_b$$

$$\overrightarrow{p}_{tot} = \overrightarrow{p}_{e^+} = \overrightarrow{p}_a + \overrightarrow{p}_b$$

Definendo con θ l'angolo fra il positrone incidente e il fotone γ_a generato, per la conservazione del momento, si ottiene

$$cos\theta = \frac{1}{pc} \left[T + \frac{2\epsilon - 1}{\epsilon} m_e c^2 \right] = \frac{\epsilon(\gamma + 1) - 1}{\epsilon\sqrt{\gamma^2 - 1}}$$

dove $\gamma = \frac{T + m_e c^2}{m_e c^2}$, mentre ϵ rappresenta la frazione di energia trasferita al fotone ed è pari a

$$\epsilon = \frac{E_a}{E_{tot}} = \frac{E_a}{T + 2m_e c^2}$$

Conoscendo il fattore ϵ e l'energia totale si può trovare allora le energie $E_a = \epsilon E_{tot}$ ed $E_b = (1 - \epsilon)E_{tot}$. L'energia trasferita al γ_a è massima quando il fotone ha la stessa direzione della particella incidente, $E_a^{Max} = \frac{E_{tot} + pc}{2}$, è minima invece quando ha la direzione opposta, $E_a^{min} = \frac{E_{tot} - pc}{2}$. Si avrà quindi

$$\epsilon_{Max} = \frac{E_a^{Max}}{E_{tot}} = \frac{1}{2} \left[1 + \sqrt{\frac{\gamma - 1}{\gamma + 1}} \right]$$

e

$$\epsilon_{min} = \frac{E_a^{min}}{E_{tot}} = \frac{1}{2} \left[1 - \sqrt{\frac{\gamma - 1}{\gamma + 1}} \right]$$

con $\gamma = \frac{T + m_e c^2}{m_e c^2}$.
La sezione d'urto differenziale, secondo la formula di Heitler, per l'annichilazione di un elettrone e un positrone è

$$\frac{d\sigma(Z, \epsilon)}{d\epsilon} = \frac{\pi r_0^2 Z}{\epsilon(\gamma - 1)} \left[1 + \frac{2\gamma}{(\gamma + 1)^2} - \epsilon - \frac{1}{\epsilon(\gamma + 1)^2} \right] \tag{1.16}$$

dove ϵ può variare fra ϵ_{Max} e ϵ_{min}.

1.2.5 Radiazione di sincrotrone

Così si definisce la radiazione emessa da un elettrone che percorre un'orbita circolare. La perdita di energia per radiazione di sincrotrone aumenta con l'aumentare dell'energia della particella ed è calcolabile classicamente attraverso la formula di Larmor[3]

$$P = \frac{2\ e^2 < a >^2}{3\ c^3}$$

La formula è valida anche per un singolo elettrone, si può quindi sostituire l'accelerazione media con quella istantanea della particella $a_{ist} = \frac{1}{m_e}\ \frac{dp}{d\tau}$. Passando poi dal sistema in quiete sull'elettrone a quello del laboratorio con le trasformate di Lorentz, si considera al posto dell'impulso $\overrightarrow{p}$ il quadrimpulso p^μ

$$\left(\frac{d\overrightarrow{p}}{d\tau}\right)^2 \Longrightarrow \left(\frac{d\overrightarrow{p}}{d\tau}\right)^2 - \frac{1}{c^2}\left(\frac{dE}{d\tau}\right)^2 = \gamma^2\left[\ \left(\frac{d\overrightarrow{p}}{dt}\right)^2 - \frac{1}{c^2}\left(\frac{dE}{dt}\right)^2\ \right]$$

con $\gamma = \frac{E}{m_e c^2}$. La sostituzione è possibile perchè la potenza dissipata $P = \frac{dE}{dt}$ è un invariante, dato che E e t hanno la stessa trasformata di Lorentz.
Così la formula di Larmor diventa:

$$P = \frac{2}{3}\frac{e^2\gamma^2}{m_e^2 c^3}\left[\left(\frac{d\overrightarrow{p}}{dt}\right)^2 - \frac{1}{c^2}\left(\frac{dE}{dt}\right)^2\ \right]$$

ed esprime la potenza irradiata da una carica in moto con il primo termine relativo all'accelerazione centripeta (variazione della direzione di $\overrightarrow{v}$), il secondo relativo a quella tangenziale (variazione del modulo di $\overrightarrow{v}$). Quando $\beta \sim 1$ il secondo termine è trascurabile, diventa invece importante il primo che interessa la luce di sincrotrone. Definendo ω_0 la velocità angolare della particella, e ricordando che $pc = \beta E$, si può sostituire

$$\left(\frac{d\overrightarrow{p}}{dt}\right)^2 = \omega_0^2 p^2 = \omega_0^2\frac{\beta^2 E^2}{c^2} = \frac{\beta^2 c^2}{R^2}\frac{\beta^2 E^2}{c^2}$$

Si calcola allora l'energia persa in un giro di periodo $T = \frac{2\pi R}{c\beta}$

$$dE = P\ T = \frac{\beta^4 E^2}{R^2}\frac{2\pi R}{c\beta} = \frac{4\pi e^2\gamma^4\beta^3}{3R} = \frac{4\pi e^2\beta^3}{3R}\ \left(\frac{E}{m_e c^2}\ \right)^4 \qquad (1.17)$$

Elettroni e positroni hanno una piccola massa a riposo, diventano velocemente relativistici e dunque sono i più adatti per la produzione di luce di sincrotrone, anche se l'effetto si può ottenere con qualsiasi particella. Nel caso elettronico possiamo scrivere

$$dE(MeV) = 8,85 \cdot 10^{-2} \frac{[E(GeV)]^4}{R(m)} \tag{1.18}$$

Unendo la (1.18) alla formula seguente

$$E(MeV/c) = p\ c = 300\ B(Tesla)\ R(m)$$

riusciamo a quantificare l'energia persa per radiazione di sincrotrone. Posto un campo magnetico $\overrightarrow{B} = 1$ T, per un elettrone di E = 1 GeV si ottiene un'orbita di R = 3,3 m e un dE = 27 keV, una perdita di energia trascurabile. Per un e^-, invece, di E = 50 GeV si ottiene R = 167 m e dE = 3,3 GeV, la perdita di energia diventa consistente.
La distribuzione spettrale del numero medio di fotoni, prodotti dalla radiazione di sincrotrone di un elettrone ultrarelativistico che si muove in un campo magnetico uniforme e costante lungo una traiettoria di lunghezza L, è data dalla relazione[2]

$$\frac{d\bar{N}}{d\omega} = \frac{\sqrt{3}\alpha}{2\pi} \left(\frac{L\gamma}{R}\right) \frac{1}{\omega_c} \int_{\frac{\omega}{\omega_c}}^{\infty} K_{\frac{5}{3}}(\eta) d\eta \tag{1.19}$$

con

- ω energia del fotone
- $\omega_c = 1,5\ \frac{\beta\gamma^3\hbar c}{R}$ energia caratteristica della radiazione di sincrotrone
- R raggio della curvatura della traiettoria
- K funzione di Macdonald

Integrando ed esplicitando la funzione di Macdonald si otiene

$$\bar{N}_{>\omega} = \frac{\sqrt{3}\alpha}{2\pi} \left(\frac{L\gamma}{R}\right) \int_0^{\infty} \frac{cosh(\frac{5t}{3})}{cosh^2(t)} e^{-\frac{\omega}{\omega_c} cosh(t)} dt \tag{1.20}$$

il numero medio di fotoni ottenuti per radiazione di sincrotrone di energia maggiore di ω. La formula si semplifica se si considera il numero medio di fotoni prodotti, cioè con energia maggiore di zero

$$\bar{N}_{>0} = \frac{\sqrt{3}\alpha}{2\pi} \left(\frac{L\gamma}{R}\right) \int_0^{\infty} \frac{cosh(\frac{5t}{3})}{cosh^2(t)} dt = \frac{5\alpha}{2\sqrt{3}} \left(\frac{L\gamma}{R}\right) \approx 10^{-2} \left(\frac{L\gamma}{R}\right) \tag{1.21}$$

Nel caso ultrarelativistico, per $\gamma \gg 1$, si ottiene $R \sim \gamma$, quindi $\bar{N}$ non dipende più dall'energia dell'elettrone, ma solo dal valore di L. Il numero medio di fotoni viene irradiato in un angolo limite dell'ordine di $1/\gamma$ attorno alla direzione della traiettoria dell'elettrone.

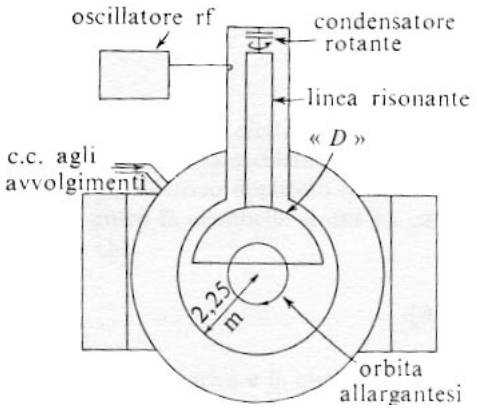


Figura 1.12: Esempio di sincrotrone a radiofrequenza.

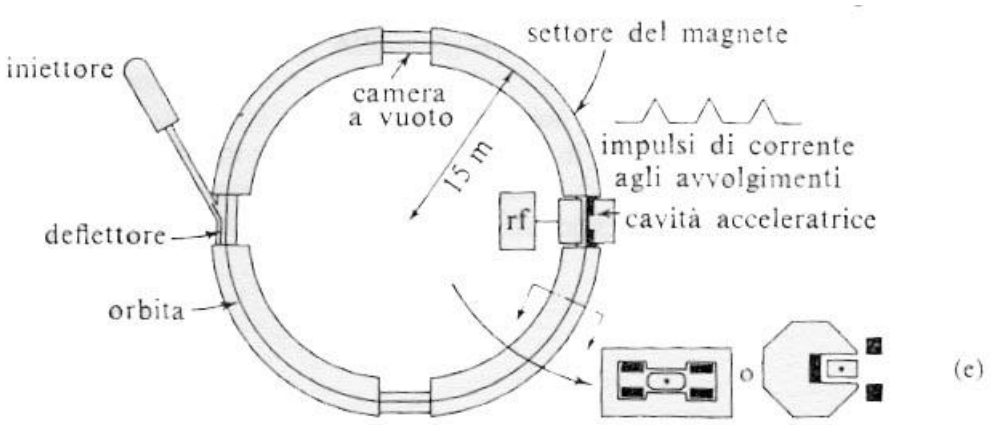


Figura 1.13: Esempio di protosincrotrone.

1.3 Adroni

Una particella carica pesante che si muove nella materia perde energia soprattutto per gli urti con gli elettroni atomici, che riescono a volte a liberarsi dal legame nucleare. La particella incidente viene dunque rallentata e la sua energia cinetica viene spesa per eccitare e ionizzare gli atomi del mezzo.
Alla perdita di energia per ionizzazione si associano altri fenomeni come la *diffusione di Rutherford* e la *diffusione multipla*, dovute ad urti elastici fra la particella incidente e il nucleo, la *radiazione Čerenkov*, quando la particella viaggia in un mezzo con velocità maggiore di quella della luce in quel mezzo, e la *radiazione di transizione*, emissione di fotoni nel passaggio fra due mezzi con costanti dielettriche differenti.

1.3.1 Ionizzazione

Data una particella di carica Ze che attraversa un materiale avente N elettroni per unità di volume, si può stimare la forza esercitata su un elettrone di un'orbita esterna dell'atomo, posto ad una distanza r, come $\frac{ze^2}{r^2}$. La traiettoria della particella pesante non è modificata in misura apprezzabile, ma la sua velocità diminuisce perchè viene trasferito un impulso Δp all'e^- pari a

$$\Delta p = \frac{2ze^2}{bv}$$

dove b è il raggio di un cilindretto centrato attorno alla traiettoria della particella incidente[1].

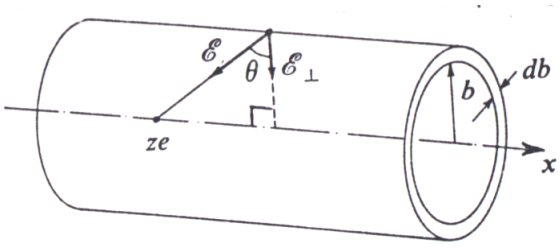


Figura 1.14: Trasferimento di impulso a un elettrone da parte di una particella carica pesante in moto.

L'energia trasferita all'elettrone è quindi:

$$dE = \frac{(\Delta p)^2}{2m} = \frac{2Z^2e^4}{mb^2v^2} \tag{1.22}$$

Per calcolare l'energia persa in un tratto dx in cui si hanno $n = 2\pi Nb\ db\ dx$ elettroni, tutti ad una distanza $(b, b+db)$ dallo ione pesante, si integra l'equazione (1.22) tra un b_{max} e un b_{min} ottenendo

$$-\frac{dE}{dx} = 2\pi N \int_{b_{min}}^{b_{max}} \frac{(\Delta p)^2}{2m} bdb = 4\pi N \frac{Z^2e^4}{mv^2} log \frac{b_{max}}{b_{min}} \tag{1.23}$$

detto potere frenante del mezzo assorbitore. Per determinare b_{max} si considerano elettroni legati alle orbite atomiche, dalla meccanica quantistica e dalla relatività si ottiene $b_{max} < \frac{\gamma v}{<\nu>}$, in cui si considera la media delle frequenze dell'atomo. Il limite per b_{min} è dato dal fatto che la distanza dell'elettrone dallo ione pesante può essere calcolata solo con l'accuratezza della sua lunghezza di de Broglie, quindi $b_{min} > \frac{\hbar}{p} = \frac{\hbar}{mv\gamma}$. L'equazione si massimizza considerando il più grande b_{max} e il più piccolo b_{min} , così viene calcolata la massima perdita di energia che la particella può avere in un tratto dx per ionizzazione.
Sostituendo si ottiene:

$$-\frac{dE}{dx} = 4\pi N \frac{Z^2e^4}{mv^2} log \frac{mv^2\gamma^2}{\hbar < \nu >}$$

Un calcolo più preciso del potere frenante è stato fatto da Bethe-Bloch con una formula semiempirica[2]

- per $T_{cut} \geq T_{max}$

$$-\frac{dE}{dx} = N_{el} \frac{Z_{inc}^2}{\beta^2} \left[ln \left(\frac{2m_e\beta^2\gamma^2 T_{max}}{i^2} \right) - 2\beta^2 - \delta - \frac{2C}{Z} \right] \tag{1.24}$$

-per $T_{cut} < T_{max}$

$$-\frac{dE}{dx} = N_{el} \frac{Z_{inc}^2}{\beta^2} \left[ln \left(\frac{2m_e\beta^2\gamma^2 T_c}{i^2} \right) - \beta^2 \left(1 + \frac{T_c}{T_{max}} \right) - \delta - \frac{2C}{Z} \right] \tag{1.25}$$

con

- T_{cut} è l'energia di taglio sopra la quale si ha produzione di raggi δ
- $T_c = min(T_{cut}, T_{max})$
- $T_{max} = \frac{2m_e(\gamma^2-1)}{1+2\gamma k+k^2}$ è l'energia massima trasferibile ad un e^- libero
- $k = \frac{m}{M}$ è il rapporto fra le masse
- N_{el} è la densità elettronica del mezzo
- $I = 16\ Z^{0,9}$ eV è il potenziale medio di ionizzazione del mezzo
- δ termine di densità, tiene conto dell'effetto di polarizzazione del mezzo dovuto al campo elettrico prodotto dalla particella incidente, diventa importante per alte energie
- $\frac{C}{Z}$ termine di shell, considera l'effetto di schermatura degli e^- interni degli atomi del mezzo, è trascurabile per le alte energie e importante per $\beta \to 0$

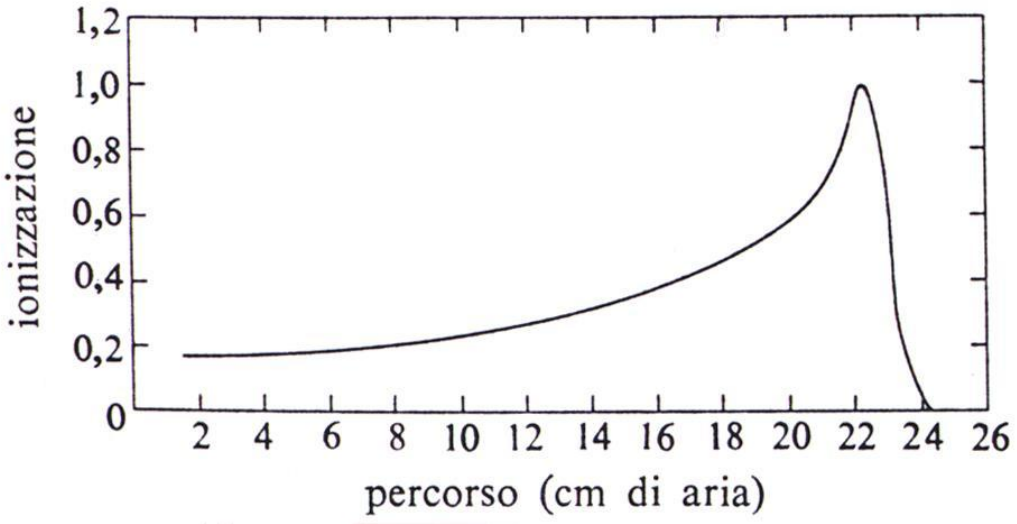


Figura 1.15: Andamento dell'energia depositata in aria da un fascio di protoni.

Il potere frenante di una particella non dipende dalla sua massa, ma dalla sua carica e dalla sua velocità. Lo studiamo in funzione di $\beta\gamma$:

- per $0 \leq \beta\gamma \leq 0,01$ la perdita di energia cresce linearmente con β, la velocità della particella è confrontabile con quella degli elettroni dell'atomo, quindi nello spostamento questa perde e acquista continuamente e^-, ottenendo una carica media minore di Z_{inc}. In questa regione la perdita di energia non è calcolabile con la formula di Bethe-Bloch, per $\beta\gamma \geq 0,01$ invece la formula è in perfetto accordo con i dati sperimentali;

- per $0,01 \leq \beta\gamma \leq 3,6$ il potere frenante decresce come $1/\beta^2$ fino ad un valore minimo ΔE_{min} in cui la particella si dice al *minimo di ionizzazione*;

- per $3,6 \leq \beta\gamma \leq 100$ la perdita di energia cresce lentamente come $log\beta\gamma$;

- per $\beta\gamma \geq 100$ siamo nel *Plateau di Fermi*, in cui $\frac{dE}{dx}$ resta costante ad un valore di energia $\Delta E_{pl} = 1,5 - 1,8\ \Delta E_{min}$ per i gas, $\Delta E_{pl} = 1,1\ \Delta E_{min}$ per i solidi.

Una particella, di massa M, entrata nel materiale con un'energia cinetica E_0, si ferma dopo aver percorso un *range*

$$R = \int_0^{E_0} \frac{dx}{dE} dE = \frac{M}{Z_{inc}} f(\beta, Z, I)$$

Se consideriamo un fascio di intensità I, il numero di particelle che si ferma cresce da un certo x_0 in poi, per fluttuazione statistica legata alla perdita di energia: questo fenomeno è chiamato *straggling*.
Oltre al range legato ad una singola particella di determinata energia che viaggia in un mezzo, viene definito anche il *range medio*: la profondità alla quale il numero di particelle non ancora ferme è del 50 % rispetto al numero iniziale.
Il *range estrapolato* è invece la profondità alla quale tutte le particelle sono ferme, si calcola come l'intersezione dell'asse delle ascisse con la tangente alla curva nel punto di range medio.

La sezione d'urto per la ionizzazione di una particella carica pesante è[2]

$$\sigma(Z, E, T_{cut}) = \frac{2\pi r_0^2 mZ}{\beta^2}\left(\frac{1 - y + \beta^2 y\ lny}{T_{cut}}\right) \qquad \text{per spin 0}$$

$$\sigma(Z, E, T_{cut}) = \frac{2\pi r_0^2 mZ}{\beta^2}\left(\frac{1 - y + \beta^2 y\ lny}{T_{cut}} + \frac{T_{max} - T_{cut}}{2E^2}\right) \qquad \text{per spin } \frac{1}{2}$$

dove $y = \frac{T_{cut}}{T_{max}}$.

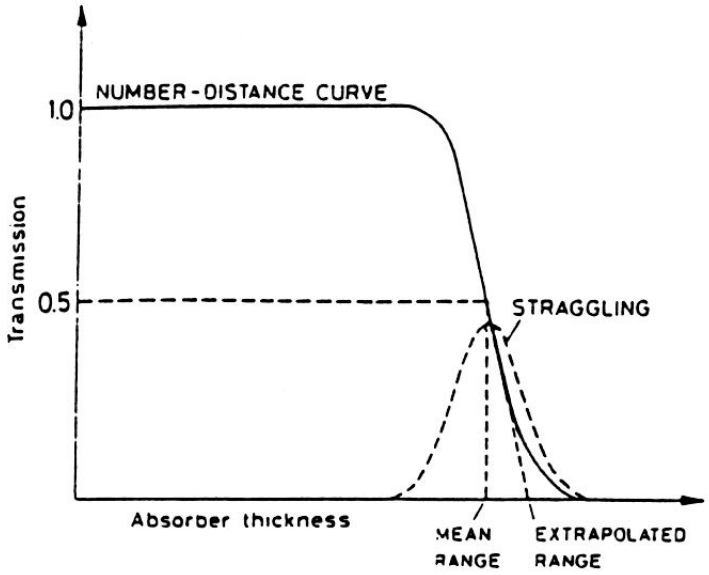


Figura 1.16: Andamento del range in funzione dello spessore del materiale attraversato.

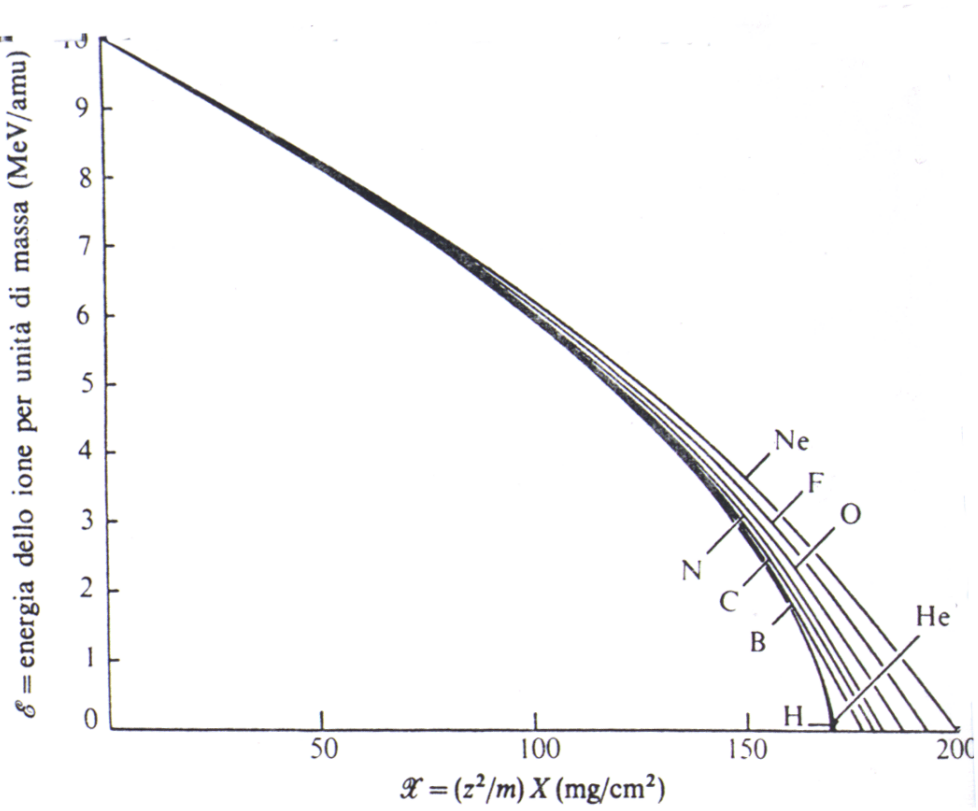


Figura 1.17: Perdita di energia per vari ioni pesanti che incidono su una lamina di Alluminio con $E_0 = 10\ MeV/amu$.

1.3.2 Diffusione di Rutherford

Una particella di carica Ze che attraversa un materiale può urtare elasticamente contro un nucleo: l'energia persa è trascurabile, mentre è influente la deflessione. La repulsione elettrostatica,infatti, varia in modo considerevole la direzione della particella incidente.
La probabilità che, attraversando uno strato dx di una sostanza con numero atomico Z, si abbia una diffusione in un angolo solido $d\omega$, causata da una deflessione θ, è

$$P d\omega = \frac{d\sigma}{d\omega} N x d\omega$$

con la sezione d'urto differenziale data dal calcolo di Rutherford

$$\frac{d\sigma}{d\omega} = \left(\frac{zZe^2}{M\beta^2 c^2}\right)^2 \frac{1}{4 \sin^4(\theta/2)}, \tag{1.26}$$

dove ze e M sono rispettivamente la carica e la massa della particella incidente, mentre Ze è la carica del mezzo attraversato[1].
La relazione (1.26) prende in considerazione solo le forze coulombiane, trascura le interazioni nucleari e le dimensioni finite del nucleo. Inoltre è ricavata con un calcolo classico, senza tener conto della meccanica quantistica. Nonostante tutte queste approssimazioni, dà buoni risultati quando la distanza minima di avvicinamento fra particella incidente e bersaglio è maggiore di $x_0 = 1,2\ A^{1/3}$ fm, in cui A è il numero di massa del bersaglio. L'equazione non è più valida per distanze minori di x_0, questo dimostra che iniziano a farsi sentire le forze nucleari.
L'angolo di diffusione medio θ_0 è

$$\theta_0 = \frac{14.1\text{MeV}}{\beta\ p}\ z\ \sqrt{\frac{x}{X_0}}\left[1 + \frac{1}{9}\ log_{10}\ \left(\frac{x}{X_0}\right)\right]$$

con p il momento della particella incidente e x/X_0 lo spessore del materiale in unità di lunghezza di radiazione.
La (1.26) comporta che sono più probabili le deflessioni a piccolo angolo, la diffusione multipla inoltre è maggiore per assorbitori di alto numero atomico, questo fa sì che proiettili più carichi e più leggeri vengano diffusi maggiormente.
Per angoli piccoli, $\theta < 1rad$, la sezione d'urto può essere scritta come

$$\frac{d\sigma}{d\omega} = \left(\frac{2zZe^2}{p\beta c}\right)^2 \frac{1}{\theta^4} \tag{1.27}$$

L'equazione è valida solo se gli angoli non sono troppo piccoli, $\theta \ll 1 rad$, quando la carica nucleare è schermata dagli elettroni atomici, questo effetto di schermo fa sì che l'equazione non diverga per $\theta \to 0$.

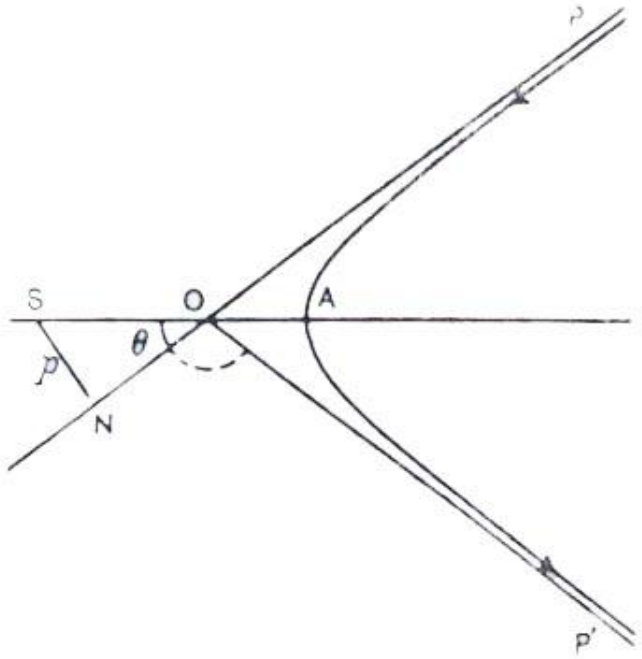


Figura 1.18: Traiettoria di una particella che ha subito una diffusione di Rutherford di angolo θ.

1.3.3 Multiple scattering

La diffusione multipla tiene conto dell'effetto cumulativo di molte piccole deflessioni nucleari che producono una deviazione θ della particella dalla sua direzione originale. Considerando uno spessore x in cui una particella subisca in media un urto che causa una deviazione dalla direzione originaria di θ_1, si avrà che le deflessioni superiori a θ_1 saranno il risultato di un solo processo di diffusione, quelle inferiori a θ_1 dovute a diffusione multipla.
Per un urto con un nucleo di carica Ze, una particella veloce di carica $Z_{inc}e$

acquista un impulso trasverso

$$\frac{\Delta p}{p} = \frac{2Z_{inc}Ze^2}{bvp} \tag{1.28}$$

Inoltre per piccole deflessioni si ha che $\frac{\Delta p}{p} = \theta$ per ogni urto. Definiamo allora θ_i gli angoli dovuti ai singoli urti e Θ la deflessione totale, con Θ_x e Θ_y le proiezioni sugli assi. Otteniamo allora:

$$\Theta^2 = \Theta_x^2 + \Theta_y^2 = \left(\sum_i \theta_i cos\phi_i\right)^2 + \left(\sum_i \theta_i sen\phi_i\right)^2$$

Si definisce la probabilità di trovare una deviazione fra Θ_x e $\Theta_x + d\Theta_x$ secondo una distribuzione gaussiana:

$$P(\Theta_x)\ d\Theta_x = \frac{e^{-\frac{\Theta_x}{<\Theta^2>}}}{\sqrt{\pi < \Theta^2 >}}\ d\Theta_x \tag{1.29}$$

dove la media sulle particelle di $< \Theta^2 > = < \sum_i \theta_i^2 > = \sum_i \bar{\theta_i^2}$ è uguale alla media sugli urti. Sostituendo θ_i con la variazione relativa dell'impulso, introducendo la (1.28) e integrando sui valori permessi del parametro d'urto b, si ottiene

$$< \Theta^2 >= 2\pi\ Nx \int_{b_{min}}^{b_{max}} \left(\frac{2Z_{inc}Ze^2}{bvp}\right) b\ db = \frac{8\pi Nx Z_{inc}^2 Z^2 e^4}{v^2 p^2} log\frac{b_{max}}{b_{min}} \tag{1.30}$$

dove N è il numero di nuclei per unità di volume e x lo spessore dell'assorbitore, che si suppone uguale al cammino della particella.
Nell'integrazione si è trascurata la variazione di v e p rispetto ad x, e la dipendenza dello Z effettivo, che tiene conto dell'effetto di schermatura del nucleo dovuto agli elettroni atomici, dal parametro b. Per il valore b_{max} la carica deve essere completamente schermata, si sceglie dunque $b_{max} = \frac{a_0}{Z^{1/3}}$, con $a_0 = 5,29 \cdot 10^{-11} m$, raggio della prima orbita di Bohr. Riguardo a b_{min}, invece, devo avere che per un singolo urto l'angolo sia piccolo rispetto ad 1 rad, dunque ottengo $b_{min} = \frac{2Z_{inc}Ze^2}{vp}$. Sostituendo ottengo:

$$< \Theta^2 >= \frac{8\pi Nx Z_{inc}^2 Z^2 e^4}{v^2 p^2} log\frac{a_0 \beta cp}{2e^2 Z_{inc} Z^{\frac{4}{3}}} \tag{1.31}$$

1.3.4 Radiazione Čerenkov

L'effetto avviene quando una carica si muove in un mezzo, con indice di rifrazione n, con velocità βc maggiore di quella della luce nel mezzo, $\frac{c}{n}$. Il fenomeno è simile alla produzione di un'onda d'urto, perchè lungo il cammino della particella viene generata un'onda elettromagnetica il cui fronte d'onda si trova sulla superficie di un cono di apertura $\alpha = sin^{-1}(1/n\beta)$.
I raggi luminosi corrispondenti allora formano un angolo con la traiettoria della particella

$$cos\theta = \frac{1}{\beta n}$$

L'indice di rifrazione n è funzione del momento p_γ dei fotoni della radiazione, viene fissato allora un limite inferiore, p_γ^{min}, tale che $n(p_\gamma^{min}) = 1/\beta$.
I fotoni vengono emessi solo se hanno un momento minore di p_γ^{max}, altrimenti sono immediatamente riassorbiti dal materiale. Tutti i fotoni della radiazione Čerenkov sono allora contenuti in un cono di apertura

$$cos\theta_{max} = \frac{1}{\beta n(p_\gamma^{max})}$$

L'intensità della luce Čerenkov dipende dal numero medio di fotoni emessi dato dalla relazione

$$\frac{dN}{dxd\nu} = \frac{2\pi z^2 e^2}{\hbar c^2} sin^2\theta = \frac{2\pi z^2 e^2}{\hbar c^2}(1 - cos^2\theta) = \frac{2\pi z^2 e^2}{\hbar c^2}\left(1 - \frac{1}{n^2\beta^2}\right) \tag{1.32}$$

Il numero di fotoni può anche essere calcolato rispetto alla variazione del momento p_γ attraverso una distribuzione Poissoniana

$$\frac{dN}{dxdp_\gamma} = \frac{2z^2 e^2}{\hbar^2 c^2}\left(1 - \frac{1}{n^2\beta^2}\right) \approx 370z^2\left(1 - \frac{1}{n^2\beta^2}\right) \tag{1.33}$$

Integrando fra i valori possibili del momento p_γ otteniamo:

$$\frac{dN}{dx} \approx 370z^2 \int_{p_\gamma^{min}}^{p_\gamma^{max}} \left(1 - \frac{1}{n^2\beta^2}\right)\, dp_\gamma = 370z^2\, p_\gamma\Big|_{p_\gamma^{min}}^{p_\gamma^{max}} - \frac{1}{\beta^2}\int_{p_\gamma^{min}}^{p_\gamma^{max}} \frac{1}{n^2(p_\gamma)}\, dp_\gamma$$

Lo spettro ottenuto è continuo e la luce è polarizzata. Una tipica radiazione Čerenkov è osservabile come luce azzurrina nell'acqua dei reattori nucleari a piscina, infatti l'effetto predomina a frequenza elevata, cioè nello spettro del visibile con tonalità tendenti all'azzurro-violetto.

1.3.5 Radiazione di transizione

Così è definita la radiazione emessa da una particella carica di energia relativistica che attraversa una superficie di discontinuità fra due mezzi differenti. Il numero medio di raggi X prodotti di energia $d\omega$ in un angolo solido $d\Omega$ intorno alla direzione del moto è

$$\frac{d^2N}{d\omega\ d\Omega} = \frac{\alpha\beta^2\sqrt{\varepsilon_2}sin^2\theta}{4\pi^2\omega}\left[\frac{1}{1-\beta\sqrt{\varepsilon_1-\varepsilon_2 sin^2\theta}} - \frac{1}{1-\beta\sqrt{\varepsilon_2}cos\theta}\right]^2 \qquad (1.34)$$

dove α è la costante di struttura fine, ε_1 e ε_2 le costanti dielettriche complesse dei due mezzi e $d\Omega = 2\pi\ sin\theta\ d\theta$. Se definiamo ω_p la frequenza di plasma elettronica[1]

$$\omega_p^2 = \frac{4\pi N_e e^2}{m_e}$$

con N_e la densità elettronica del mezzo, possiamo calcolare il numero di fotoni nel caso ultrarelativistico in cui la radiazione è concentrata in una piccola regione di angolo $\theta \sim \frac{1}{\gamma} \ll 1$

$$\frac{d^2N}{d\omega\ d\chi} = \frac{\alpha\chi}{\pi\omega}\left[\frac{1}{\gamma^{-2}+\eta^{(1)}+\chi} - \frac{1}{\gamma^{-2}-\eta^{(2)}+\chi}\right]^2 \qquad (1.35)$$

dove η indica il rapporto $\eta^{(i)} = \left(\frac{\omega_p^{(i)}}{\omega}\right)^2$ e $\chi = 2(1-cos\theta) \sim \theta^2 \ll 1$.
Se integro la (1.35) rispetto all'energia ottengo la densità angolare dei raggi X prodotti al passaggio della particella dal mezzo 1 al mezzo 2:

$$\frac{d\bar{N}}{d\chi} = \frac{\alpha\ \chi}{\pi}[A+B+C]$$

dove i coefficienti A, B e C sono:

$$A = \frac{c^2}{2}\left[\frac{1}{a^2(x^2+a^2)} + \frac{1}{a^4}ln\frac{x^2}{x^2+a^2}\right]_{x_2}^{x_1}$$

[1]Plasma: quarto stato della materia, costituito da un insieme di particelle con cariche elettriche positive e negative, dotate di grande mobilità. La caratteristica di un plasma è che il numero di particelle cariche positivamente eguaglia quello delle particelle di carica opposta, in modo che il fluido appare macroscopicamente neutro. Lo stato di plasma è caratterizzato dal numero di particelle per unità di volume e dalla temperatura. La frequenza delle oscillazioni presenti in un plasma è detta *fraquenza di plasma*, per gli elettroni vale $\nu_p = 8,98\cdot10^3\ \sqrt{nHz}$,con la densità n espressa in cm^{-3}.

$$B = -\frac{cd}{a^2 - b^2}\left[\frac{1}{b^2} ln \frac{x^2}{x^2 + b^2} - \frac{1}{a^2} ln \frac{x^2}{x^2 + a^2}\right]_{x_2}^{x_1}$$

$$C = \frac{d^2}{2}\left[\frac{1}{b^2(x^2 + b^2)} + \frac{1}{b^4} ln \frac{x^2}{x^2 + b^2}\right]_{x_2}^{x_1}$$

con i coefficienti ausiliari:

- $x_i = \frac{1}{\omega_i}$
- $\sigma(\gamma, \chi) = \chi + \frac{1}{\gamma^2}$
- $c = \left(\frac{1}{\omega_p^{(1)}}\right)^2$
- $d = \left(\frac{1}{\omega_p^{(2)}}\right)^2$
- $a^2 = \sigma c$
- $b^2 = \sigma d$

La densità spettrale dei fotoni generati è ricavata invece integrando la (1.35) rispetto all'angolo χ ottenendo

$$\frac{d\bar{N}}{d\omega} = \frac{\alpha}{\pi\omega}\left[\frac{t + u}{t - u} ln \frac{\omega + u}{\omega + t} + \frac{t}{\omega + t} + \frac{u}{\omega + u}\right]_0^{\chi_{max}}$$

con i coefficienti:

- $t = \eta^{(1)} + \frac{1}{\gamma^2}$
- $u = \eta^{(2)} + \frac{1}{\gamma^2}$

Definito $\omega_c^{(i)} = \gamma\, \omega_p^{(i)}$ l'energia caratteristica dei fotoni prodotti nel mezzo i, per $\omega < \omega_c^{(2)} \ll \omega_c^{(1)}$ ottengo

$$\bar{N}(\omega) = \frac{\alpha}{\pi}\left[2 ln \frac{\omega_c^{(1)}}{\omega}\left(ln \frac{\omega_c^{(1)}}{\omega_c^{(2)}} - 1\right) - ln^2 \frac{\omega_c^{(1)}}{\omega_c^{(2)}} + 2\, ln 2 - \frac{\pi^2}{24}\right]$$

il numero medio di γ generati da una particella ultrarelativistica che attraversa la giunzione fra due mezzi differenti.

Capitolo 2

La radioterapia

In questo capitolo vengono presentati i fondamenti fisici della radioterapia e le sue grandezze caratteristiche, i fasci e gli acceleratori utilizzati per i piani di trattamento e i danni che possono derivare dall'esposizione a radiazioni. Sono inoltre analizzate alcune delle nuove terapie in via di sviluppo, che utilizzano adroni e ioni leggeri.

Introduzione

Le radiazioni ionizzanti causano effetti biologici, al loro passaggio nel tessuto umano, e possono quindi essere utilizzate per la cura di vari processi morbosi. Questo è lo scopo della radioterapia: cedere energia nella zona tumorale in modo da distruggere le cellule malate, ma con precisione e accuratezza per preservare quelle sane. Possono essere usati diversi tipi di raggi: radiazioni emesse da sostanze radioattive naturali o artificiali, raggi X o fasci di particelle, elettroni o adroni, accelerati. La scelta dipende dal tipo e dalla locazione del tumore da trattare.

2.1 Basi della radioterapia

Le radiazioni vengono solitamente divise in *poco penetranti*, a bassa energia, e *molto penetranti*, ad alta energia. Le prime utilizzano fasci di fotoni, resi omogenei attraverso l'utilizzo di filtri, in modo da circoscrivere con precisione la zona da irradiare. Se il processo patologico da trattare è superficiale e di limitata esten-

sione, si utilizzano radiazioni poco energetiche, che vengono assorbite soprattutto nei primi strati della cute, evitando così di danneggiare le strutture sottostanti. Quando il focolaio è situato invece in profondità, per concentrare su di esso una dose elevata ed evitare danno negli organi circostanti, si utilizzano fasci da diverse angolazioni o fasci rotanti, così da irradiare una volta sola i tessuti sani e più volte la zona tumorale.
Poco penetranti sono anche considerate le radiazioni prodotte da elementi radioattivi o da isotopi artificiali[4]. Queste sostanze vengono inserite in aghi di platino che vengono infissi direttamente nella zona da trattare (*rad. interstiziale*) oppure vengono incorporate in particolari materiali plastici che vengono poi modellati sulla superficie cutanea del paziente. Un altro metodo che utilizza la radioattività naturale o artificiale è la *teleradioterapia*, in cui la sostanza è posta a distanza dall'organismo, racchiusa in involucri isolanti a pareti di piombo, capaci di arrestare quasi tutte le radiazioni. Un'apertura lascia passare il fascio da utilizzare nel trattamento. Questa tecnica viene usata soprattutto con il Co^{60} (*cobaltoterapia*). Un altro utilizzo dei radioisotopi è l'introduzione di essi in forma liquida nell'organismo, attraverso le cavità sierose, per via endolinfatica o endovenosa. L'elemento radioattivo viene concentrato in particolari tessuti nei quali si esplica l'effetto delle radiazioni emesse, ad esempio lo iodio I^{131} si concentra nella tiroide, il fosforo P^{32} nel tessuto osseo.

Figura 2.1: Modello Theratron-80, Theratronics, utilizzato in cobaltoterapia.

Le radiazioni ad alta energia vengono invece prodotte con acceleratori lineari, costituiti da un tubo a vuoto in cui è accelerato un fascio di elettroni. Questo, urtando contro un anodo, provoca l'emissione di radiazioni di energia fino a 20 MeV, con un potere di penetrazione elevato che permette di raggiungere focolai profondi. Si ottiene così una distribuzione più omogenea della radiazione fra i vari tipi di tessuto e una minore dispersione laterale.
Recentemente l'interesse si è spostato sull'utilizzo di particelle più pesanti, dotate di maggiore energia, come protoni, neutroni veloci e ioni quali l'elio, il carbonio, l'ossigeno e il neon. Si è sviluppata così, a fianco della radioterapia, la *adroterapia* che utilizza *adroni*, cioè particelle pesanti soggette all'interazione forte. La terapia con adroni presenta grandi vantaggi rispetto a quella convenzionale, sia per la precisione maggiore nell'irradiare una zona tumorale, sia per la maggiore efficacia del trattamento, ma la ricerca in questo settore deve essere ancora approfondita.

2.1.1 Acceleratori per la radioterapia

L'acceleratore maggiormente usato per la terapia con fotoni è il LINAC, acceleratore lineare a radiofrequenza composto da elettrodi in cui è presente un campo elettrico oscillante di frequenza tale che la particella sia sempre accelerata. Quindi questa riceve incrementi di energia quando attraversa i tubi in cui c'è il campo, viaggia invece a velocità costante negli spazi vuoti che intervallano gli elettrodi. Se si considera un campo di frequenza fissa ω, si dovrà aumentare la lunghezza dei tubi in cui passa il fascio man mano che la velocità delle particelle cresce, perchè nello stesso periodo di tempo percorrerà spazi maggiori. Energie elevate si raggiungono dunque solo su distanze molto grandi, ne è un esempio l'acceleratore lineare per elettroni da 25 GeV dell'università di Stanford che raggiunge una lunghezza di 2 miglia.
L'energia a cui è possibile accelerare una particella è esprimibile attraverso parametri caratteristici dell'acceleratore come potenza P, lunghezza d'onda λ e lunghezza totale L:

$$E = \frac{k\sqrt{PL}}{\lambda^{1/4}}$$

Il *range energetico* di questi acceleratori è solitamente compreso fra 3 e 25 MeV: quelli la cui energia massima arriva a 6 MeV sono utilizzati solo per la terapia con raggi X, altri permettono di selezionare l'energia del fascio terapeutico offrendo

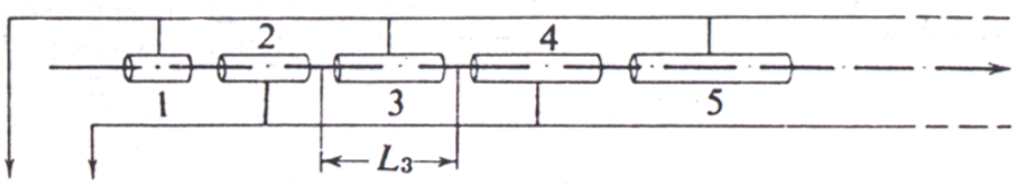


Figura 2.2: Schema di un acceleratore lineare del tipo Lawrence-Sloan. I tubi sono di lunghezza crescente, in particolare il terzo ha una lunghezza $L_3 = \sqrt{\frac{6eV_0}{m}}\ \frac{T}{2}$, con T il periodo del campo oscillante applicato.

dei valori discreti di energia, come l'$SL\ 75-20$ in cui sono possibili fasci di 5, 6, 8, 10, 12, 14, 17 e 20 MeV.

In questo lavoro di tesi è stato utilizzato un acceleratore lineare che termina con una testata rotante, *gantry*, e permette di effettuare un'irradiazione isocentrica, cioè irradiare la zona tumorale facendo ruotare il fascio attorno al paziente, così da depositare una dose maggiore nella massa malata.

Figura 2.3: Esempio di acceleratore lineare rotante, in grado di irradiare in modo dinamico, cioè anche quando è in movimento.

Nella testata dell'acceleratore il fascio di fotoni viene prodotto, per radiazione di bremsstrahlung, inviando elettroni accelerati su un bersaglio. I raggi γ passano poi in un magnete per eliminare ogni possibile contaminazione del fascio primario.

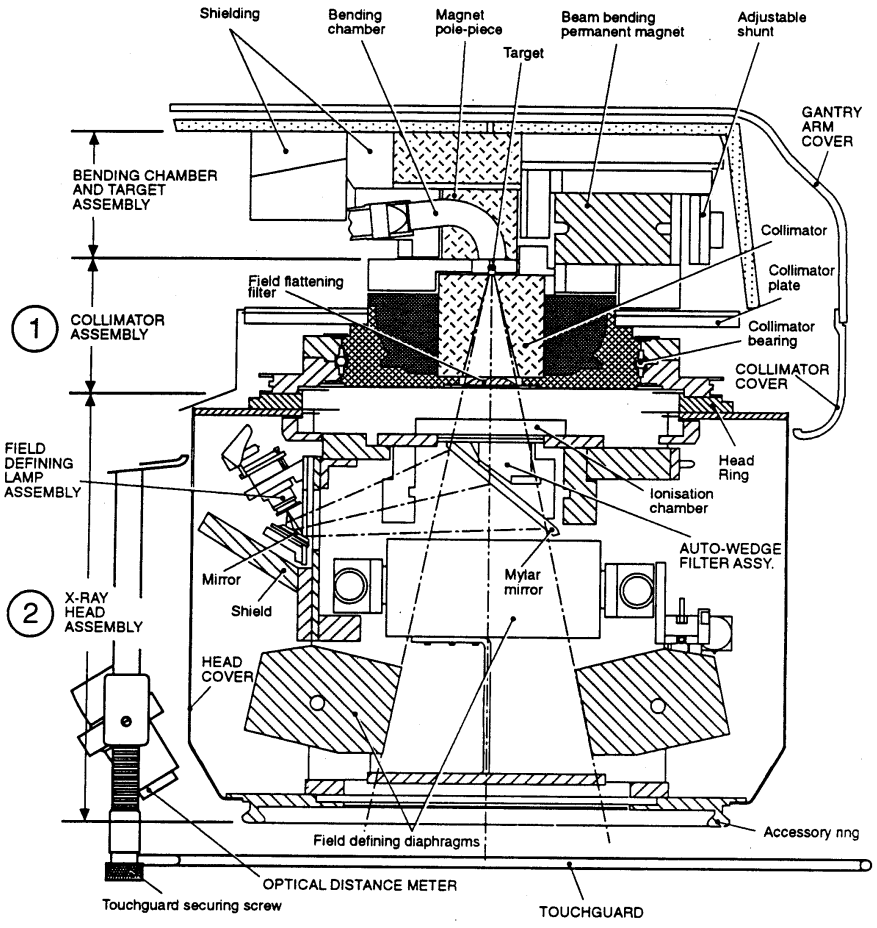


Figura 2.4: Schema della testata di un acceleratore lineare rotante per fasci terapeutici di fotoni.

Nella testata è presente un **collimatore primario**[5], composto da un cilindro di tungsteno con un foro conico che definisce l'apertura del fascio. La grandezza dello spot focale è di 4 mm e il semiangolo del fascio pari a 14^0. A contatto con il foro conico troviamo il **filtro omogenizzatore** che rende uniforme la distribuzione di intensità all'interno del campo di radiazione.
Il campo è monitorato da una **camera a ionizzazione**, posizionata dopo il filtro, divisa in tre unità: Dose1, Dose2 e Slave. Ognuna di queste camerette è costituita da due facce, collettore e polarizzatore, separate da un anello, ed è riempita di aria o azoto secco. I collettori sono completamente isolati, i polarizzatori sono invece collegati ad un potenziale di $-300V$. Dose1 e Dose2 forniscono una misura di dose su due canali, mentre Slave, posizionato fra le due camerette precedenti, corregge eventuali non uniformità del campo.
Dopo la camera a ionizzazione è situato un **filtro cuneiforme** che permette di ottenere l'effetto di un campo della stessa forma. Il filtro viene posizionato da un motore elettrico a 60^0 all'inizio del trattamento e spostato verso angoli minori dopo che una percentuale fissata della dose totale è stata rilasciata.
Nella testata è presente anche un **collimatore secondario**, detto diaframma, che definisce le dimensioni del campo attraverso controlli ottici e meccanici. Infine è presente il **sistema ruotante di diaframmi**, costituito da 4 diaframmi indipendenti che definiscono un campo rettangolare. Sono controllati elettricamente a coppie e, per ogni posizione della testata, sono sempre bilanciati in modo che le loro facce interne siano allineate con la sorgente di raggi X, questo limita al minimo la penombra del fascio. La posizione dei diaframmi è monitorata da una coppia di potenziometri di precisione.
Per centrare correttamente il fascio sulla zona da irradiare si utilizza un **sistema ottico di definizione**, posizionato a lato dei diaframmi. E' costituito da una lampada che, attraverso un sistema di specchi, proietta un raggio di luce nell'apertura definita dai diaframmi, su di essa è posta anche una griglia che proietta l'immagine di una croce e di un triangolo, per segnalare il centro del fascio e l'orientamento del filtro cuneiforme rispetto al campo.
Un **secondo sistema ottico** indica la distanza tra la sorgente di radiazione e il paziente, detta SSD (Source Skin Distance). Una sorgente luminosa, attraverso una serie di lenti, proietta sulla cute del paziente una scala: la distanza cercata si ottiene quando il centro del campo coincide con la scala proiettata.

Figura 2.5: Acceleratore lineare, tipo MEVATRON K-P2, modello PRIMUS, prodotto dalla Siemens, utilizzato nella Divisione di Radioterapia all'ospedale San Giovanni Antica Sede di Torino.

2.1.2 Piani di trattamento

L'obbiettivo della radioterapia è un controllo della massa tumorale tale che questa non si diffonda ulteriormente, ma retroceda. Lo scopo si raggiunge somministrando al paziente dosi adeguate tali da uccidere le cellule tumorali, ma non arrecare danni irreversibili ai tessuti sani. Per ogni neoplasia occorre dunque preparare un piano di trattamento personalizzato, dopo aver identificato con cura il volume da trattare, il tipo e lo stadio della massa tumorale e, naturalmente, le condizioni del paziente.
Per preparare i piani di trattamento vengono analizzate le curve dose-effetto che rappresentano la possibilità di distruggere il tessuto tumorale in funzione della dose assorbita[1], e la possibilità di arrecare danni ai tessuti sani. Si nota dal grafico seguente che alla certezza di distruggere la massa tumorale si accompagna una probabilità del 100% di arrecare danni agli organi circostanti sani. Per ottenere un giusto compromesso viene allora definito il *rapporto terapeutico*, cioè il rapporto fra la dose corrispondente ad una probabilità del 50% di causare danni e quella relativa al 50% di controllare il tumore.

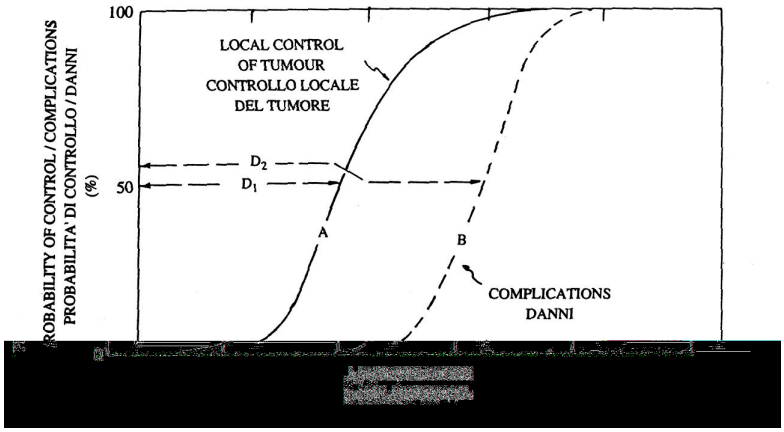


Figura 2.6: Curve dose - effetto per tessuti neoplastici e normali, indicano rispettivamente la dose necessaria per il controllo del tumore e la probabilità di causare danni.

[1]Dose: grandezza radioterapica fondamentale, è il rapporto fra l'energia media ceduta dalla radiazione ad un volume e la massa di tale volume.

In un piano di trattamento intervengono diversi parametri, quali la *conformità* o selettività balistica, cioè la differenza di dose al bersaglio e ai tessuti sani, la *radiosensibilità* delle cellule, cioè come il paziente risponde all'irragiamento, e questo dipende anche dalla locazione del tumore, e infine il *frazionamento* della dose, cioè la durata e il numero delle sedute. Importante è soprattutto definire con precisione il volume da trattare e modellare la distribuzione di dose su di esso. Con questo scopo si è sviluppata la terapia conformazionale che, attraverso sistemi di sagomatura e modulazione del fascio, determina la dose ai confini del *planning target volume* detto PTV. Attualmente il trattamento conformazionale con fasci convenzionali di fotoni si ottiene utilizzando collimatori ad apertura variabile, detti *multileaf collimator*, composti da coppie di lamelle che si spostano in modo indipendente fra loro, controllate da un calcolatore, e che permettono di sagomare il fascio secondo la forma del tumore così come questo è visto dalla direzione di incidenza.
Per creare un buon piano di trattamento occorre valutare entro determinati limiti di accuratezza la dose di riferimento relativa al caso patologico in questione, valutare l'omogeneità della distribuzione di dose, soprattutto ai bordi del PTV e infine effettuare un controllo in vivo in vari punti di riferimento durante il piano di trattamento. Per il monitoraggio in tempo reale della dose rilasciata sono necessari dosimetri particolari, come la camera monitor a pixel, oggetto di studio in questo lavoro di tesi.

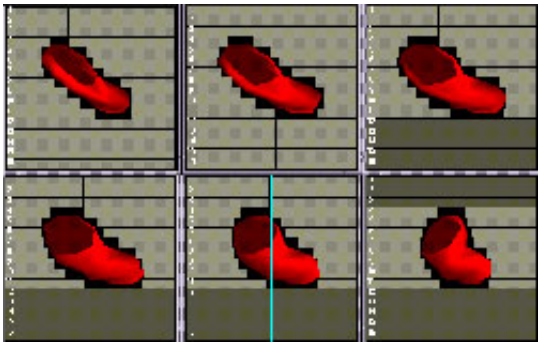


Figura 2.7: Esempi di come viene ottimizzata l'irradiazione attraverso il collimatore multilamellare.

2.1.3 Effetti sull'uomo

Lo studio degli effetti delle radiazioni sull'uomo è molto complesso perchè è complessa la stessa struttura dell'organismo umano. I vari organi di cui esso è composto sono infatti dati dalla combinazione di più tessuti con caratteristiche differenti e funzioni specifiche proprie. Ogni tessuto è costituito da unità elementari, cellule, formate principalmente da idrogeno, ossigeno, carbonio e azoto. La radiazione che attraversa le cellule viventi eccita gli atomi e le molecole della struttura cellulare, dando luogo a frammenti dotati di carica elettrica, ioni e radicali liberi, che sono instabili e decadendo interagiscono con i nuclei. L'effetto totale è comunque funzione della dose assorbita e si manifesta come un danno più o meno grave alla cellula, ma che può causarne anche la morte. I danni meno gravi vengono riparati per rigenerazione della cellula o per sostituzione delle cellule danneggiate attraverso la mitosi di quelle sane.
Le cellule si differenziano fra loro a seconda della funzione che hanno nell'organismo, e quindi reagiscono anche in modo diverso alle radiazioni.

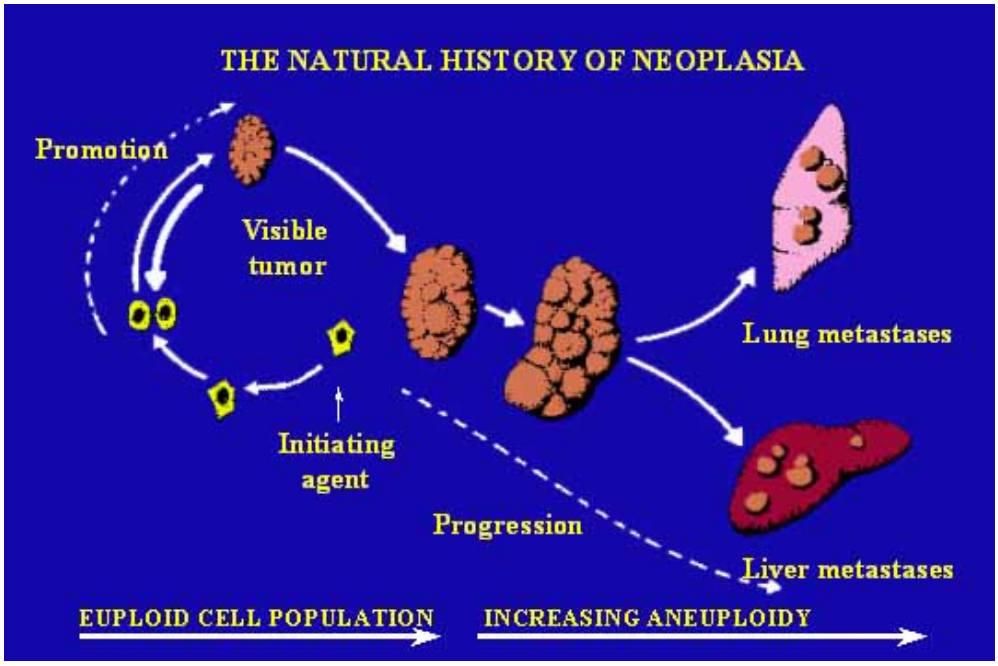


Figura 2.8: Sviluppo di una neoplasia.

Tabella VI. - Tempo approssimativo di sopravvivenza e modalità di morte nei roditori panirradiati

Dose al corpo intero (Gy)	*Tempo approssimativo di morte dopo panirradiazione*	*Meccanismo di morte*
100 e oltre	Da pochi minuti a 48 ore	Sindrome del sistema nervoso centrale
10-100	3-5 giorni	Sindrome gastrointestinale
2-10	10-30 giorni	Sindrome del midollo osseo
Inferiore a 2	Poche settimane prima degli animali di controllo non irradiati	Accorciamento della durata della vita

Figura 2.9: Tabella riportante i tempi approssimativi di sopravvivenza e le modalità di morte in roditori panirradiati, cioè irradiati su tutto il corpo. La panirradiazione si verifica nel caso di incidenti nucleari, più raramente si utilizza a scopo terapeutico per la cura di leucemie acute, con dosi molto basse, essendo in questo caso la dose letale per l'uomo di soli 500 rad.

Definiamo allora la *radiosensibilità* di una cellula come la proprietà di venir più o meno danneggiati dall'esposizione a radiazioni ionizzanti [4], in generale questa è maggiore quanto maggiore è l'attività riproduttiva delle cellule che compongono il volume irradiato. La radioterapia ha dunque tanto più successo quando maggiore è la differenza fra la radiosensibilità del tessuto neoplastico e quella dei tessuti sani circostanti, questa differenza è detta *intervallo terapeutico*. La radiosensibilità è inversamente proporzionale al grado di differenziazione e direttamente proporzionale alla capacità di riprodursi. Dunque le cellule con più alta attività metabolica, come quelle tumorali, sono più sensibili alle radiazioni, questo porta dei danni a tessuti vitali, come il midollo osseo, le ghiandole linfatiche e gli organi riproduttivi, mentre sono meno radiosensibili i muscoli e le ossa. La cellule in via di sviluppo, inoltre, possono essere più facilmente danneggiate di quelle già completamente formate. La radiosensibilità aumenta quando la massa irradiata è ricca di ossigeno, e questo è un inconveniente, essendone i tessuti neoplastici poveri. Si cerca allora di aumentare il grado di ossigenazione nel tessuto, posizionando il paziente in una camera ad ossigeno o attraverso metodi farmacologici.

Danni somatici ed ereditari

Gli effetti sull'uomo si differenziano in somatici, se si manifestano nell'individuo esposto, e ereditari, se colpiscono invece la prole.
I danni somatici dipendono principalmente dalla quantità e dalla intensità della dose assorbita, cioè dal modo in cui la dose è stata accumulata e da come questo accumulo sia stato frazionato nel tempo. Importante è anche il tipo di radiazioni utilizzato, perchè ad uguali dosi non sempre corrispondono gli stessi effetti, l'efficacia biologica [2] è diversa a seconda del tipo di particelle e delle condizioni di irragiamento.
Gli effetti somatici si dividono in **immediati** e **tardivi** a seconda dell'intervallo di tempo fra l'esposizione e la loro comparsa. Possono essere lievi, come rossore, bruciature ed epilazione, o gravi, come la sindrome acuta da radiazioni, per dosi assorbite di alcuni Gy, causata da gravi danni al sistema nervoso centrale. Per dosi molto elevate tutti i casi di esposizione comportano la morte. Viene allora definita nell'intervallo di possibile sopravvivenza la *dose letale media*, la dose per la quale il 50% dei soggetti esposti soccombe. Per l'uomo questa dose è intorno ai $3 - 5\ Gy$. I principali effetti tardivi sono costituiti dal danno genetico, come la sterilità, o dall'induzione a neoplasie, cataratta, fibrosi da radiazioni o danni ai vasi sanguigni.

Induzione al cancro

Sembra che le radiazioni siano in grado di causare la comparsa di tumori nei tessuti che sono stati direttamente esposti, ma è difficile valutare l'entità del rischio. Le teorie al riguardo maggiormente confermate sono quelle che considerano la mutazione somatica e la teoria virale[6]. La prima ipotizza che il DNA di una cellula possa essere alterato in modo tale da modificare l'informazione in esso contenuta. Se una cellula "difettosa" è in grado di dividersi indipendentemente dal meccanismo di controllo dell'organismo, essa è considerata una cellula cancerosa. Le radiazioni danneggiano il DNA in vari modi, è dunque possibile che avvengano delle mutazioni nelle cellule somatiche che danno così origine a neoplasie. Questa teoria della mutazione somatica rappresenta attualmente il meccanismo più accettato per spiegare l'induzione al cancro. La teoria virale invece sostiene

[2]Efficacia biologica: capacità di provocare danni per una determinata dose

che molti virus sono associati ad una maggiore probabilità di cancerogenesi e che le radiazioni possono stimolare un virus cancerogeno silente. E' noto che l'irragiamento di tessuti blocca la risposta immunitaria, quindi il soggetto può essere maggiormente esposto all'attacco di un virus cancerogeno.
Sia la teoria delle mutazioni somatiche sia quella virale implicano alterazioni intracellulari, ma nessuna delle due è in grado di dimostrare se il danno indotto dalla radiazione all'esterno della cellula può o no provocare il cancro.
Il cancro radioindotto è stato studiato fin dal 1902, ma con molti problemi per quanto riguarda la valutazione del rischio, soprattutto perchè il periodo di latenza tra l'esposizione e la comparsa del tumore può essere di decenni, questo comporta lunghi periodi di controllo. Inoltre, studiando persone deliberatamente o accidentalmente esposte a radiazioni, spesso la determinazione della dose è imprecisa o i livelli di dose possono non essere adeguati alla valutazione del rischio. Esistono anche problemi di tipo statistico legati alla necessità di osservare estese popolazioni per lunghi periodi di tempo, allo scopo di evidenziare aumenti nella frequenza di casi di tumore indotto da radiazioni.

2.2 Misure in radioterapia

Per misurare l'energia depositata nei tessuti e dunque poter calcolare i rischi di un'esposizione a radiazioni, sono state introdotte nella radioterapia delle grandezze fondamentali quali la dose, quantità di energia assorbita da un corpo per unità di massa, il LET, trasferimento lineare di energia, l'RBE, efficacia biologica relativa e l'OER, percentuale di ossigeno nei tessuti irradiati. Analizziamo ora la definizione e il significato di queste grandezze radioterapiche.

2.2.1 Dose

La dose assorbita è la grandezza di maggiore interesse in radioterapia. E' definita come il rapporto fra l'energia media assorbita da un corpo sottoposto a radiazioni e la sua massa:

$$D = \lim_{m \to 0} \frac{\bar{E}}{m}$$

L'unità di misura utilizzata per il calcolo della dose nel SI è il Gray (Gy), definito come $J \cdot kg^{-1}$, anche se nella pratica si utilizza il *rad* pari a $10^{-2}\ Gy$. La

massa considerata nel calcolo della dose deve essere abbastanza piccola da non influenzare troppo il rapporto considerato, così che si possano stabilire delle dosi limite per determinati intervalli di m, ma abbastanza grande da ottenere un valore finito. L'energia considerata invece è data dalla differenza fra l'energia delle particelle che entrano e quella delle particelle che escono.

Vengono utilizzate anche altre definizioni legate al concetto di dose:
- **dose equivalente**: pari alla dose assorbita moltiplicata per un coefficiente, detto *fattore di qualità*, che dipende dall'efficacia biologica abbinata al tipo di radiazione utilizzata (ad esempio per i fotoni e gli elettroni il fattore di qualità è 1, per i neutroni è 10, per le particelle α è 20). La dose equivalente è misurata in sievert.
- **dose equivalente efficace**: data dalla dose equivalente assorbita in un tessuto moltiplicata per il *fattore ponderale di rischio* legato al tipo di tessuto considerato. Anch'essa viene misurata in sievert.

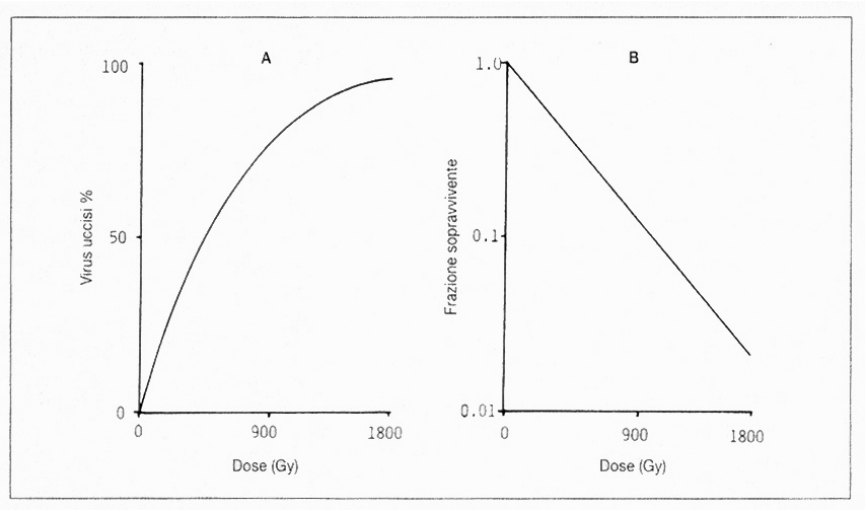


Figura 2.10: **A**) Diagramma lineare della percentuale del danno in funzione della dose di radiazioni espressa in Gy. **B**) Diagramma semilogaritmico della percentuale di sopravvivenza in funzione della dose.

2.2.2 LET

In radiobiologia è fondamentale considerare l'energia depositata dalla particella localmente. Viene quindi introdotta una nuova grandezza dosimetrica, il Trasferimento Lineare di Energia, indicato con L_∞, definito come il rapporto tra l'energia ΔE depositata da una particella carica e il percorso effettuato Δx. Il LET è misurato in keV/μm, dipende in modo direttamente proporzionale dalla carica. Rappresenta la densità di energia ceduta dalle particelle lungo il percorso, e dunque collegato al valore di dose rilasciata localmente.

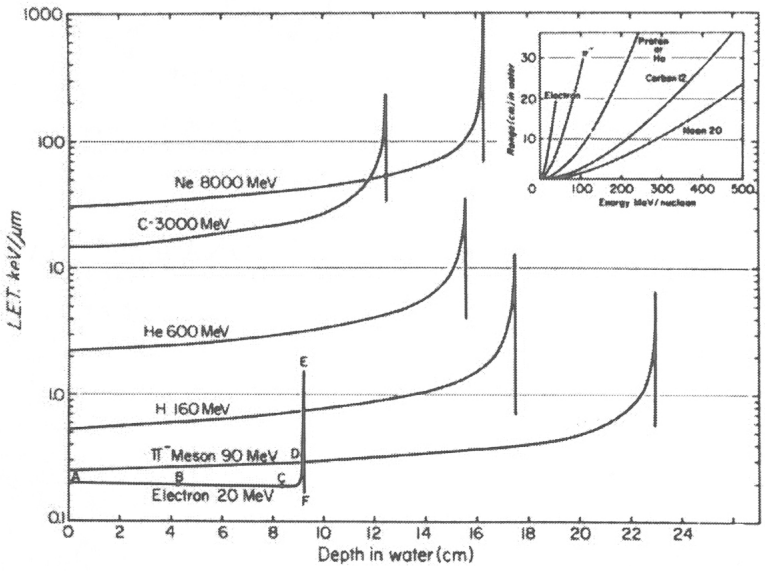


Figura 2.11: Curve di Bragg per particelle cariche in acqua. E' mostrato l'andamento del LET in funzione della profondità: all'ingresso della particella in acqua in suo valore è basso, raggiunge invece il massimo subito prima dell'arresto della particella. Si nota che la curva riguardante gli elettroni è inadeguata poichè non vengono considerati i cambiamenti di direzione durante l'interazione con le molecole di acqua. L'inserto in alto mostra l'andamento del range in funzione dell'energia per le medesime particelle.

Particella	Carica	Energia (MeV)	L_∞ (keV/µm)
Elettrone	-1	0.01	2.30
		0.1	0.42
		1.0	0.25
Fotone	0	γ del Cobalto-60 fotoni da 4-25 MV	0.2 - 2
Protoni	+1	2	16
		5	8
		10	4
		200	0.4
Particelle $\alpha = He^{+2}$	+2	5	95
Neutroni	0	5	3-30
Ioni di carbonio = C^{+6}	+6	10-250 MeV/u [a]	170 - 14

(a) L'energia cinetica dei nuclei è solitamente espressa dividendo l'energia cinetica *totale* espressa in MeV per la massa dell'atomo misurata in unità di massa atomica. Il suo simbolo è MeV/u. In pratica questo rapporto E_N è uguale all'energia cinetica di ciascun protone e neutrone del nucleo e viene detta di solito "*energia in MeV per nucleone*".

Figura 2.12: Valori del LET delle particelle ionizzanti e delle radiazioni utilizzate in radioterapia. Si nota che gli ioni carbonio hanno un LET 100 volte maggiore di quello dei fasci convenzionali di fotoni.

2.2.3 RBE

Non essendo la dose assorbita un buon indicatore degli effetti che le radiazioni hanno sui tessuti biologici, è stato introdotto l'RBE, RadioBiological Equivalent, per misurare i danni provocati a parità di dose da radiazioni differenti. Viene definito come il rapporto fra la dose di una radiazione standard per produrre un determinato effetto e la dose della radiazione in esame necessaria per produrre il medesimo effetto. La radiazione standard generalmente utilizzata è costituita da un fascio di raggi X di energia pari a 200 keV.
Possiamo inoltre definire l'RBE come un rapporto di dosi che producono lo stesso livello di sopravvivenza. Esso cresce con il LET fino ad un valore massimo sufficiente ad uccidere la cellula, quindi valori ancora superiori del LET non possono essere espressi biologicamente perchè la morte cellulare è già stata raggiunta, l'energia depositata in eccesso va dunque persa e si raggiunge uno stato di saturazione detto *overkill*. Importante è il tipo di tessuto irradiato, l'RBE infatti è massimo nei tessuti radioresistenti, minore in quelli radiosensibili.

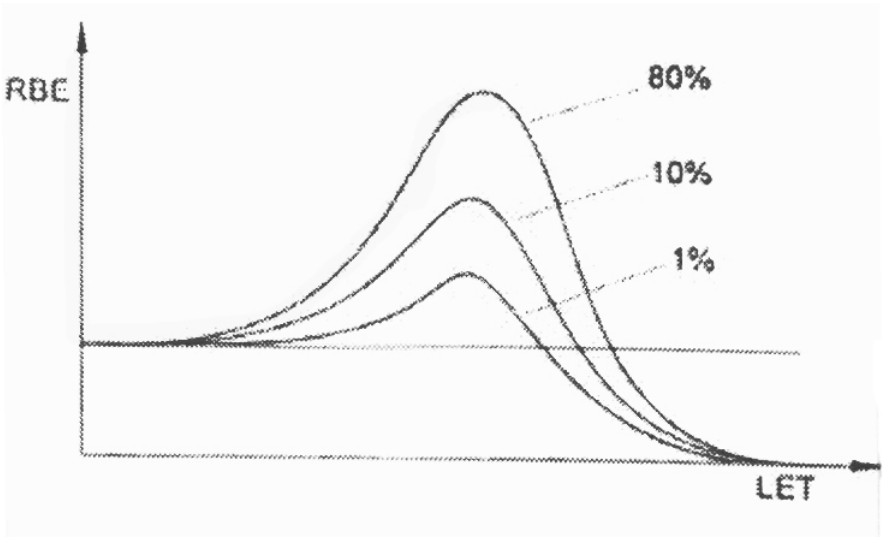


Figura 2.13: Rappresentazione schematica del RBE in funzione del LET per diversi valori del livello di sopravvivenza.

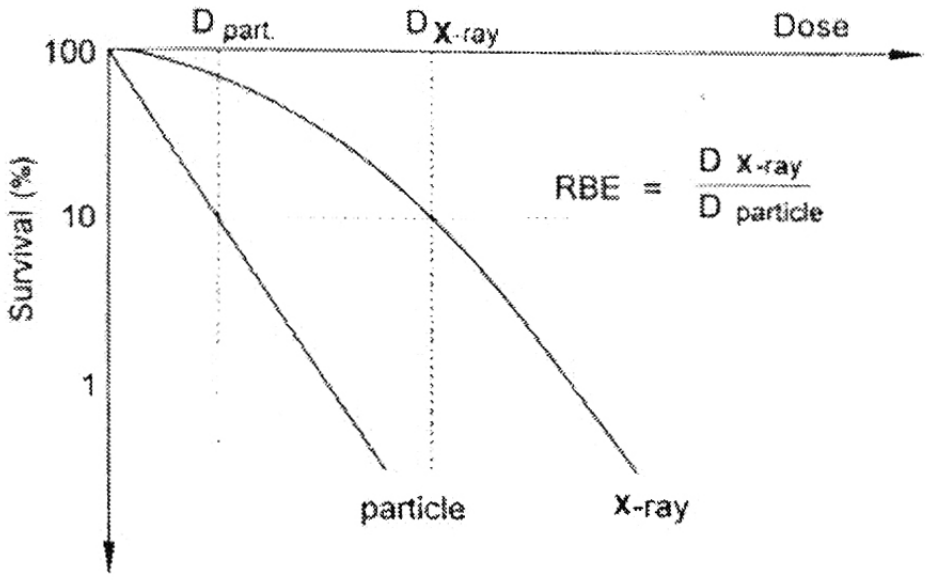


Figura 2.14: Curve schematiche di sopravvivenza cellulare.

2.2.4 OER

Un altro parametro fondamentale nella valutazione degli effetti biologici della radiazione è il contenuto di ossigeno nei tessuti irradiati, è stato dunque definito l'OER, *Oxygen Enhancement Ratio*, come il rapporto tra la dose necessaria ha provocare un dato effetto biologico in assenza di O_2 e quella necessaria per provocare lo stesso effetto nell'aria ad 1 atm. Si considera questo confronto fra tessuti poveri o ricchi di ossigeno perchè questo elemento ha una funzione sensibilizzante. A riguardo esistono differenti teorie: alcune attribuiscono l'aumento dei danni in presenza di ossigeno alla produzione di radicali liberi, altre invece affermano che le proprietà radiosensibilizzanti dell'ossigeno siano legate alla sua affinità elettronica che porta l'ossigeno a reagire con gli elettroni liberi prodotti dalla ionizzazione, impedendone così la ricombinazione.

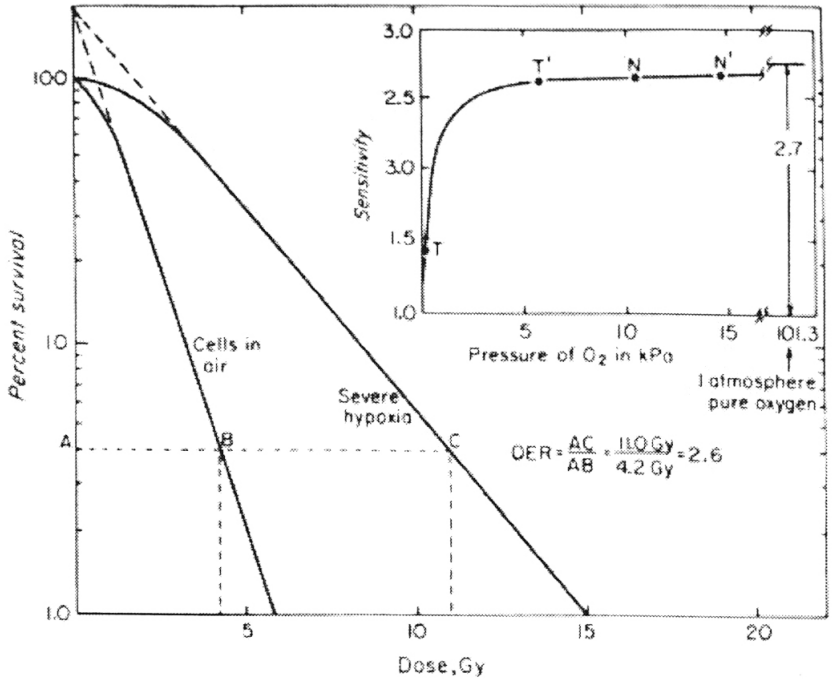


Figura 2.15: Percentuale di sopravvivenza per cellule irradiate in presenza e in assenza di O_2. Si nota che le cellule irradiate in aria sono circa 3 volte più radiosensibili di quelle in condizioni di ipossia, cioè con grave carenza di ossigeno. L'inserto in alto mostra la dipendenza logaritmica della radiosensibilità dalla pressione.

2.3 Sviluppo di nuovi trattamenti

Nella situazione attuale l'unico mezzo valido per combattere i tumori maligni è riuscire ad effettuare una diagnosi in tempo, quindi agli studi nel campo della terapia si affiancano ricerche per migliorare la diagnostica. Negli ultimi 20 anni l'incidenza delle neoplasie nel mondo è notevolmente aumentata, di conseguenza si è anche sviluppata la ricerca finalizzata a ridurre il numero dei decessi e ad allungare la vita del paziente. Attualmente in Europa il 45% dei pazienti viene *guarito*, ha cioè un periodo di sopravvivenza superiore ai 5 anni. Il 90% delle guarigioni dipende dal controllo loco-regionale del tumore primario[7].
I trattamenti tuttora esistenti sono:

- la rimozione chirurgica del tumore, quando questo non ha prodotto metastasi e non è in organi vitali, utilizzata nel 22% dei casi;

- la radioterapia, quando il tumore è di piccole dimensioni e ben localizzato, utilizzata nel 12% dei casi;

- la chemioterapia, che utilizza farmaci per ostacolare la moltiplicazione delle cellule tumorali maligne, cura il 5% dei casi;

- la immunoterapia, che stimola le difese dell'organismo iniettandovi delle sostanze ricche di anticorpi, è utilizzata nel 5% dei casi.

Sovente queste tecniche terapeutiche vengono combinate fra loro per sfruttare i vantaggi e porre rimedio ai limiti di ciascuna di esse. Di fondamentale importanza è dunque una diagnosi precoce e approfondita, serve dunque sensibilizzare la popolazione al riguardo e migliorare le tecniche diagnostiche esistenti. Naturalmente bisogna anche migliorare i trattamenti locali, in modo da evitare cure poco efficaci e riuscire a trattare con successo tumori di difficile localizzazione e tumori radioresistenti alla terapia convenzionale. Occorre anche un miglioramento dei trattamenti sistematici per le metastasi, così da far fronte al problema delle diagnosi tardive.

2.3.1 Radioterapia con fotoni

Lo studio oggi della radioterapia tende ad ottimizzare determinati parametri per ottenere il maggior risultato nella cura dei tumori con il minimo danno dei tessuti sani, cercando di controllare localmente la neoplasia. I fattori presi in considerazione sono:

- la radiosensibilità,
- il recupero cellulare,
- la riossigenazione dei tessuti,
- la ripopolazione di cellule sane,
- la ridistribuzione delle cellule.

Non esistono grandi differenze tra la **radiosensibilità** delle cellule sane e di quelle tumorali, così come non differisce di molto la velocità di riparazione dei due tessuti. Ma essendo quelli neoplastici poveri di ossigeno, da cui dipende il **recupero cellulare**, il danno accumulato nel volume tumorale per dosi multiple è maggiore di quello subito dai tessuti sani. Naturalmente deve essere studiato con attenzione il frazionamento della dose in base al caso patologico in questione. Solitamente vengono effettuati dei cicli standard di dosi giornaliere di 2 Gy, per 5 giorni la settimana, per 4-6 settimane.
La **riossigenazione** avviene in gran parte dei tessuti neoplastici e non invece in quelli sani. La distanza di diffusione dell'ossigeno nei tessuti è di circa 150 μm, quindi tutte le cellule di diametro maggiore diventano anossiche[3] dopo aver metabolizzato tutto l'ossigeno disponibile. Dopo l'irradiazione le cellule più ricche di O_2 sono morte perché più radiosensibili, rimangono solo quelle ipossiche. Eliminate le cellule uccise, la massa tumorale si riduce, ma i vasi sanguigni del tumore, che non sono molto interessati dall'irradiazione, continuano a portare le stesse quantità di ossigeno. Così improvvisamente le cellule malate diventano ben ossigenate e quindi radiosensibili a dosi successive. Purtroppo sono ancora scarse le conoscenze sulla riossigenazione, sui tempi caratteristici e sui parametri che la interessano.

[3]Anossia: condizione patologica delle cellule dovuta a mancanza di ossigeno. Quando l'anossia è localizzata in un particolare tessuto i danni dipendono dalla sensibilità di questo, in alcuni casi si arriva alla morte cellulare, si parla allora di *ischemia*. Quando invece si tratta solo di una diminuzione dell'ossigeno disponibile per i processi respiratori cellulari si parla di *ipossia*

La **ricrescita dei tessuti** si verifica nell'intervallo fra due frazionamenti, ma si ha una maggior proliferazione nei tessuti neoplastici che in quelli sani. Esiste anche la possibilità di una **ridistribuzione delle cellule** in un ciclo cellulare, in cui queste variano la loro radiosensibilità. Infatti esistono fasi nello sviluppo di una cellula in cui questa è più vulnerabile, si può dunque studiare il periodo d'irradiazione in funzione dello sviluppo cellulare del tumore.

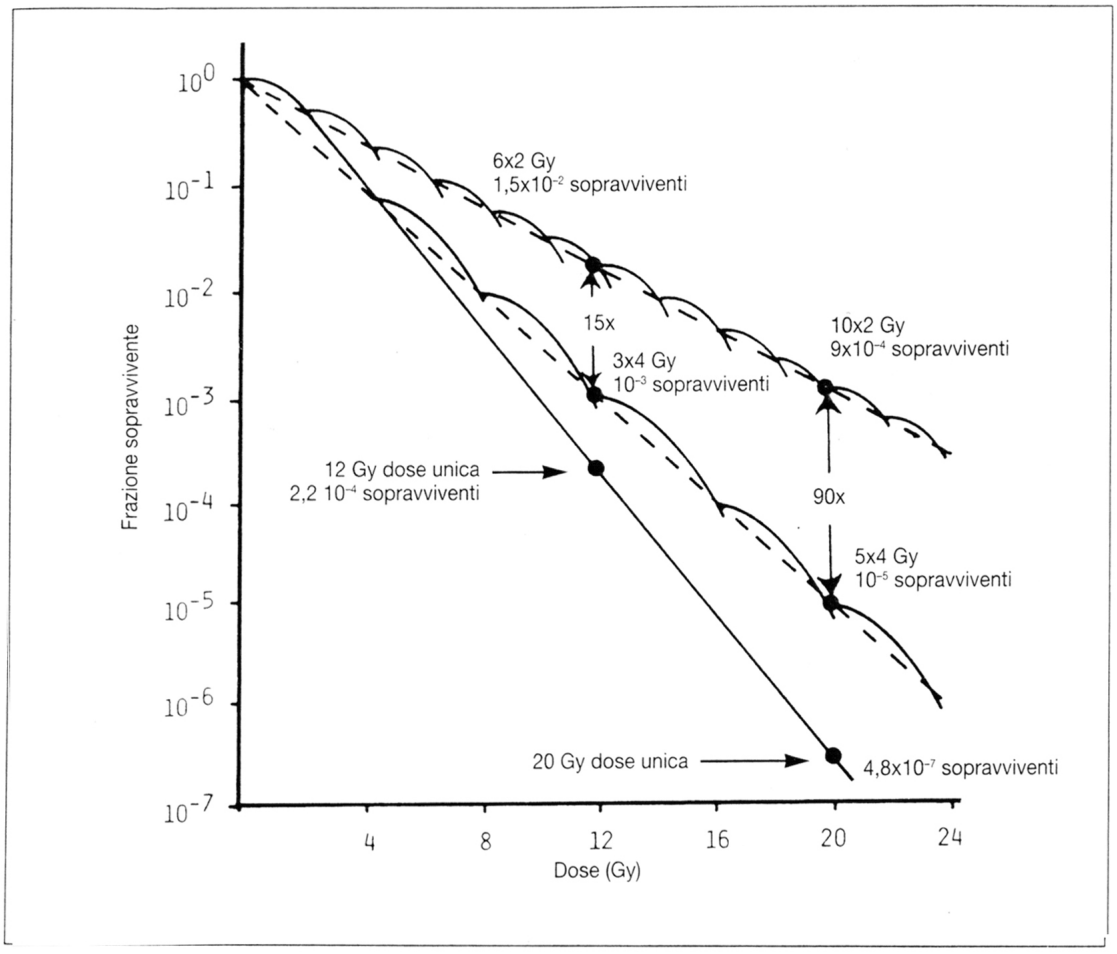


Figura 2.16: Diagramma della sopravvivenza cellulare con dosi uniche di 12 e 20 Gy, con frazioni di 4 Gy e con frazioni di 2 Gy per ogni dose.

2.3.2 Adroterapia

La terapia con raggi γ presenta degli svantaggi: i fotoni infatti, anche a basse energie, hanno una diffusione laterale non trascurabile e la distribuzione della dose rispetto alla profondità non è abbastanza selettiva a causa dell'assorbimento di tipo esponenziale. Per far fronte a questo problema si è presa in considerazione la possibilità di utilizzare altre particelle, come i protoni, che avessero una migliore accuratezza. Si è così sviluppata la adroterapia che utilizza particelle adroniche, soprattutto protoni e nuclei di atomi leggeri.

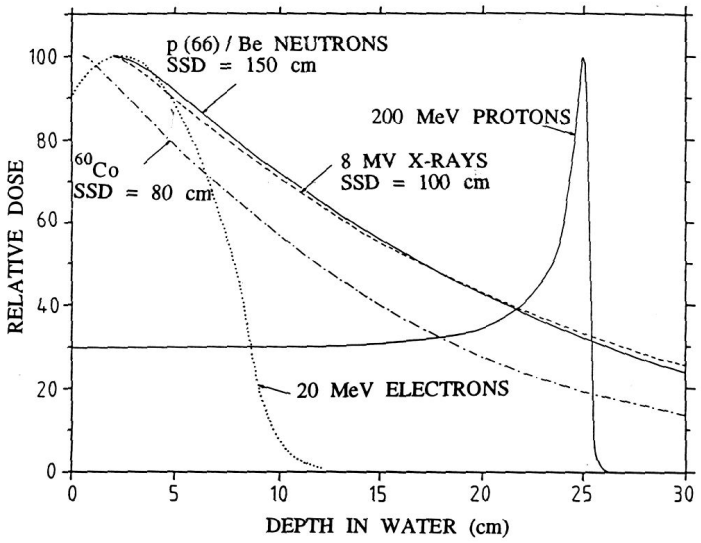


Figura 2.17: Perdita di energia in acqua per diversi tipi di particelle. Si nota la differenza fra la deposizione di energia dei fotoni e dei protoni, questo permette di utilizzare differenti particelle a seconda della locazione della massa neoplastica.

I neutroni e gli ioni leggeri sono particelle ad alto LET, cioè depositano energia con una densità elevata, i protoni, invece, a basse profondità si comportano come particelle a basso LET (elettroni e fotoni), ma al termine del percorso il loro LET aumenta improvvisamente. Dunque il trattamento con adroni risulta molto più efficace della radioterapia convenzionale. La curva dose - profondità dei protoni mostra chiaramente l'andamento dell'energia depositata e la posizione del picco

di Bragg, che deve coincidere con la posizione del tumore (come rappresenta la fig. 2.17).
Questo picco è determinato dal fatto che la particella, rallentando nel mezzo, perde più energia verso la fine del suo percorso. Oltre il picco la dose rilasciata decresce bruscamente e si ha una coda fino a che la particella non si ferma. La posizione in profondità del picco dipende dall'energia iniziale del protone e dal materiale attraversato[4]. La dimensione della sezione trasversale invece dipende dalla dispersione laterale del fascio. E' possibile allargare in profondità il picco variando l'energia del fascio durante l'irradiazione in modo da sovrapporre picchi diversi e ottenerne uno allargato, SOBP (*Spread Out Bragg Peak*). Questo si ottiene anche interponendo uno spessore variabile di materiale assorbitore fra la sorgente e il bersaglio. Esiste anche la possibilità di controllare la distribuzione laterale dei fasci focalizzando, deflettendo e direzionando le particelle attraverso campi magnetici perpendicolari alla direzione di propagazione. I protoni utilizzati in radioterapia hanno un'energia fra $60 - 70\ MeV$ o $200 - 250\ MeV$, che corrisponde rispettivamente ad un percorso di $2.5 - 3\ cm$ o $25 - 30\ cm$. L'adroterapia costituisce quindi una tecnica terapeutica ad alta precisione, indispensabile nella cura di neoplasie situate in prossimità di organi critici che non devono assolutamente essere irradiati.

2.3.3 Radioterapia con ioni leggeri

Fasci di ioni leggeri, come Carbonio, Ossigeno e Neon, hanno diffusioni laterali trascurabili e depositano la maggior percentuale di energia alla fine del loro percorso. Questo permette di ottenere una definita distribuzione di dose in profondità, migliore anche di quella dei protoni. Per la loro intensa ionizzazione locale sono efficaci per i tumori radioresistenti. Oltre il picco, però, la coda è più accentuata rispetto ai protoni, per il fenomeno della frammentazione, cioè della disgregazione di un nucleo pesante in frammenti più leggeri in seguito ad urti con gli atomi del mezzo. Questi frammenti, avendo massa minore, hanno un percorso più lungo e vanno ad aumentare la dose rilasciata oltre il picco.
Con ioni leggeri sono già stati trattati casi di tumore a Berkeley, in California, mentre in Giappone è in costruzione un impianto interamente dedicato a questo scopo[7].

[4] Vedi in proposito l'appendice A sulla simulazione con protoni da 62 MeV in materiali aventi differenti densità.

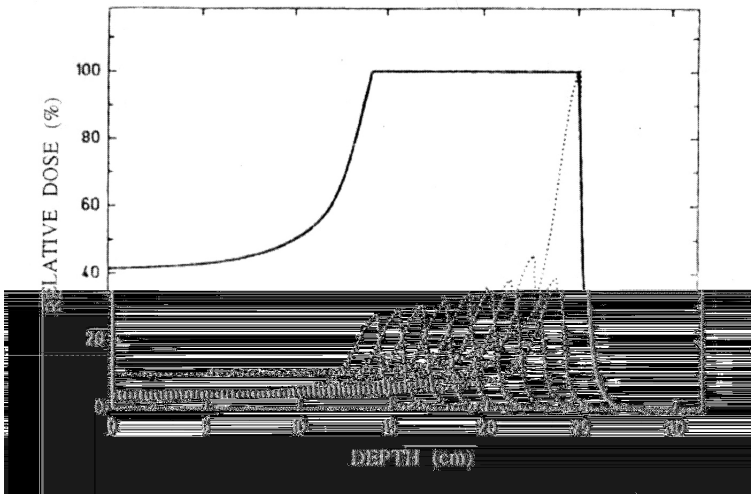


Figura 2.18: SOBP per protoni, raggiunto con la sovrapposizione di numerosi picchi di Bragg.

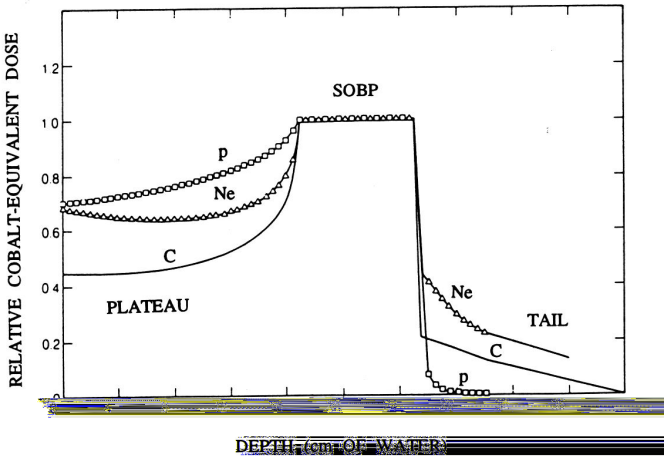


Figura 2.19: SOBP per protoni e ioni leggeri, Carbonio e Neon. Notiamo una minor dose rilasciata a basse profondità da parte degli ioni rispetto ai protoni, ma una coda maggiore dopo il picco di Bragg.

2.3.4 BNCT

Alcuni composti del Boro possono essere accumulati in certi tessuti tumorali, soprattutto nel cervello. In seguito all'irradiazione con neutroni, gli atomi del boro catturano i neutroni generando un atomo di Litio e una particella α. Queste particelle hanno un percorso nel tessuto inferiore al millimetro, rilasciano dunque tutta la loro energia nel tumore. Si è sviluppata così la *Boron Neutron Capture Therapy*. Con questa tecnica, a parità di dose nei tessuti circostanti, l'energia rilasciata nella massa tumorale è più grande di un terzo. I primi trattamenti sono stati fatti in Giappone, e mostrano quanto questa terapia sia indicata per neoplasie al cervello, ma si inizia a studiare i risultati anche per i trattamenti di altri tipi di tumore.

2.3.5 Progetto TERA

In Italia non sono ancora presenti centri dove venga utilizzata l'adroterapia. A questo proposito è nata la fondazione TERA, TErapia con Radiazioni Adroniche, una Fondazione per l'Adroterapia Oncologica con lo scopo di introdurre per l'anno 2006 anche nel nostro Paese queste nuove tecniche, soprattutto quelle utilizzanti i protoni, con la costruzione di un centro a Milano. Al progetto TERA partecipano enti di ricerca italiani ed esteri, tra cui la sezione di Novara, quella di Ginevra, presso il CERN, l'università di Milano Bicocca e il Dipartimento di Fisica delle università di Genova e Torino.

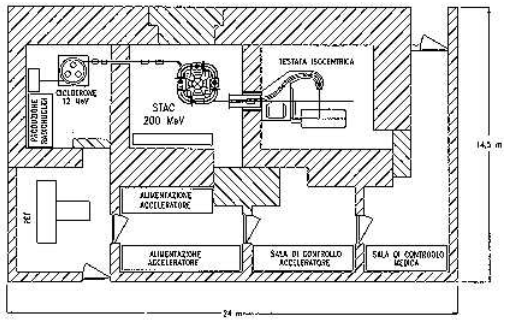


Figura 2.20: Schema della possibile struttura di un futuro Centro di Protonterapia che utilizzerà uno STAC, Sincrotrone Tecnologicamente Avanzato e Compatto, e un ciclotrone da 12 MeV, usato sia come iniettore che per produrre radioisotopi per la PET.

Capitolo 3

Geant4

In questo capitolo viene spiegata la struttura gerarchica di Geant4, il codice di simulazione utilizzato in questo lavoro di tesi. Vengono analizzati i principali dominii e le classi necessarie all'implementazione di un programma di simulazione. Infine viene analizzato un file macro, necessario all'avvio di un run, e i comandi già disponibili.

Introduzione

Geant4 è un codice di simulazione che sfrutta la programmazione ad oggetti, progettato per rendere il più trasparente possibile l'implementazione dei processi fisici e permettere di ottenere risultati affidabili. E' un toolkit basato sul metodo Monte Carlo, nato per la simulazione di rivelatori nella fisica delle alte energia, ma sfruttato anche in altri settori, ad esempio per applicazioni mediche, spaziali o nucleari. Questo codice è stato utilizzato per sviluppare il programma di simulazione della camera monitor a pixel, per fasci di fotoni a bassa energia, studio di questa tesi.

3.1 La storia di Geant4

Il progetto Geant4 è stato approvato dal *Detector Research and Developement Committee*, DRDC, del Cern alla fine del '94. Un prototipo è stato presentato alla fine del '95, si è poi iniziata allora la vera e propria implementazione del codice, con la prima versione α nella primavera del '97, a cui seguì la versione β

nella metà del '98. Un pacchetto di simulazione più completo dei precedenti fu presentato nel dicembre '99, Geant4.1.1, a cui seguirono continui aggiornamenti fino all'ultima versione, Geant4.3.2, del luglio 2001[8].
Il software di Geant4 è stato sviluppato grazie alla collaborazione di circa 100 scienziati, provenienti da tutto il mondo. Hanno collaborato più di 40 istituti, fra Europa, Russia, Giappone, Canada e Stati Uniti. Importante il contributo del CERN di Ginevra che ha collaborato con l'Istituto Nazionale di Fisica Nucleare (INFN) di Bari, Genova, Milano, Padova, Roma e Torino[12].

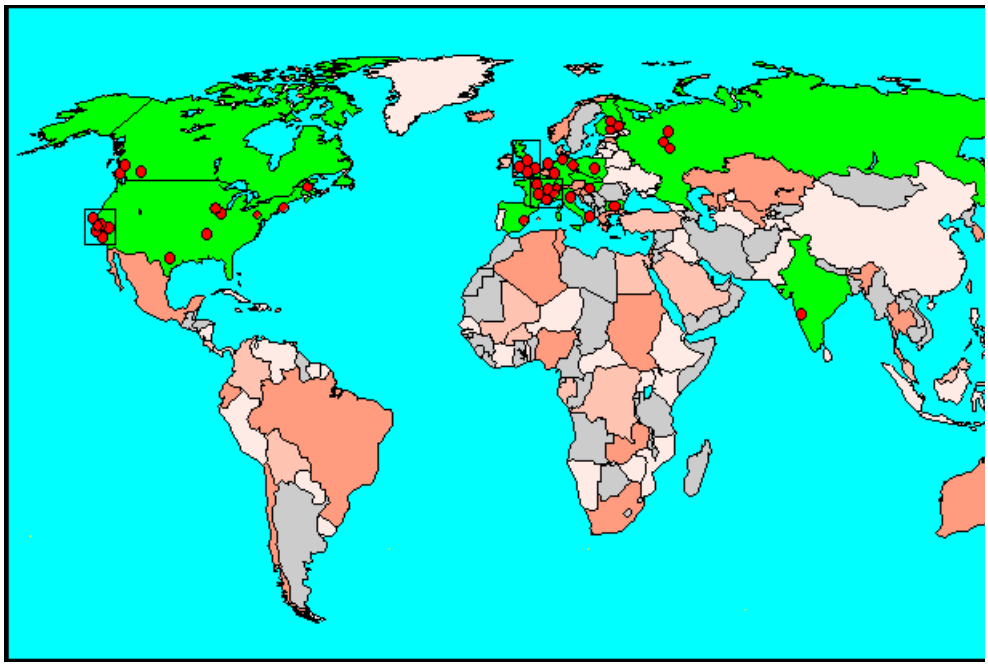


Figura 3.1: Mappa delle zone in cui viene utilizzato Geant e dei centri che hanno contribuito alla sua realizzazione.

3.2 La struttura di Geant4

Questo programma di simulazione permette di descrivere la geometria e i materiali che compongono il rivelatore e le caratteristiche del fascio. Quindi il codice genera un certo numero di eventi, inserito dall'utente tramite il comando */run/beamOn*. Ad ogni evento viene generata una particella con una traiettoria

dipendente dall'impulso iniziale, dai campi magnetici che incontra e dalle interazioni che avvengono quando questa attraversa la materia. Il codice divide la traiettoria in step, la cui lunghezza dipende dal materiale attraversato, e per ognuno di essi calcola l'energia persa dalla particella. Al termine del run, dopo aver generato le n particelle richieste, viene sommata l'energia persa nel volume sensibile in ogni evento. Si può ottenere dunque un unico valore di energia, espresso in MeV, come nel caso di un volume sensibile cilindrico di aria, o un istogramma, se il volume sensibile viene diviso in voxel e l'energia viene sommata in funzione del voxel di appartenenza, come nel caso della camera monitor a pixel.
Geant può essere integrato con altri software: OPENGL e DAWN, ad esempio, forniscono il supporto grafico per il rendering della geometria del rivelatore e la visualizzazione delle tracce delle particelle, CLHEP mette a disposizione elementi matematici quali l'istogramma, la n-tupla, i vettori e le distribuzioni random, fondamento di una simulazione MonteCarlo.
Geant è diviso in dominii e sottodominii, indipendenti fra loro, riguardanti i vari aspetti della simulazione. Questa struttura rende possibile l'implementazione multipla di diversi processi fisici e permette all'utente di personalizzare ed estendere il toolkit in tutti i dominii. Questi contengono classi strettamente collegate fra di loro che possono essere richiamate e ridefinite dall'utente.
Facciamo un breve sommario dei dominii di Geant:

- **Geometry** è il dominio legato alla geometria del rivelatore e ai materiali che lo costituiscono, contiene classi che permettono di definire dei solidi, di relazionarli fra loro e posizionarli nello spazio.

- **Particle e Material** gestiscono la definizione delle particelle e dei materiali utilizzati. Contengono le tabelle di un numero di isotopi definiti (*G4IsotopeTable*), di tutte le particelle fino ad ora conosciute (*G4ParticleTable*), di alcuni elementi (*G4ElementTable*) e di alcuni materiali (*G4MaterialTable*) implementati nella simulazione.

- **Processes** è il dominio che gestisce tutti i processi fisici che partecipano alle interazioni della particella con la materia. Contiene l'implementazione di tutti i modelli fisici, divisi in base al range, al tipo di particella e al materiale che questa attraversa. In modo trasparente vengono calcolate le sezioni d'urto del processo fisico che interviene nell'interazione, o attraverso il calcolo analitico o utilizzando dei tabulati di dati sperimentali.

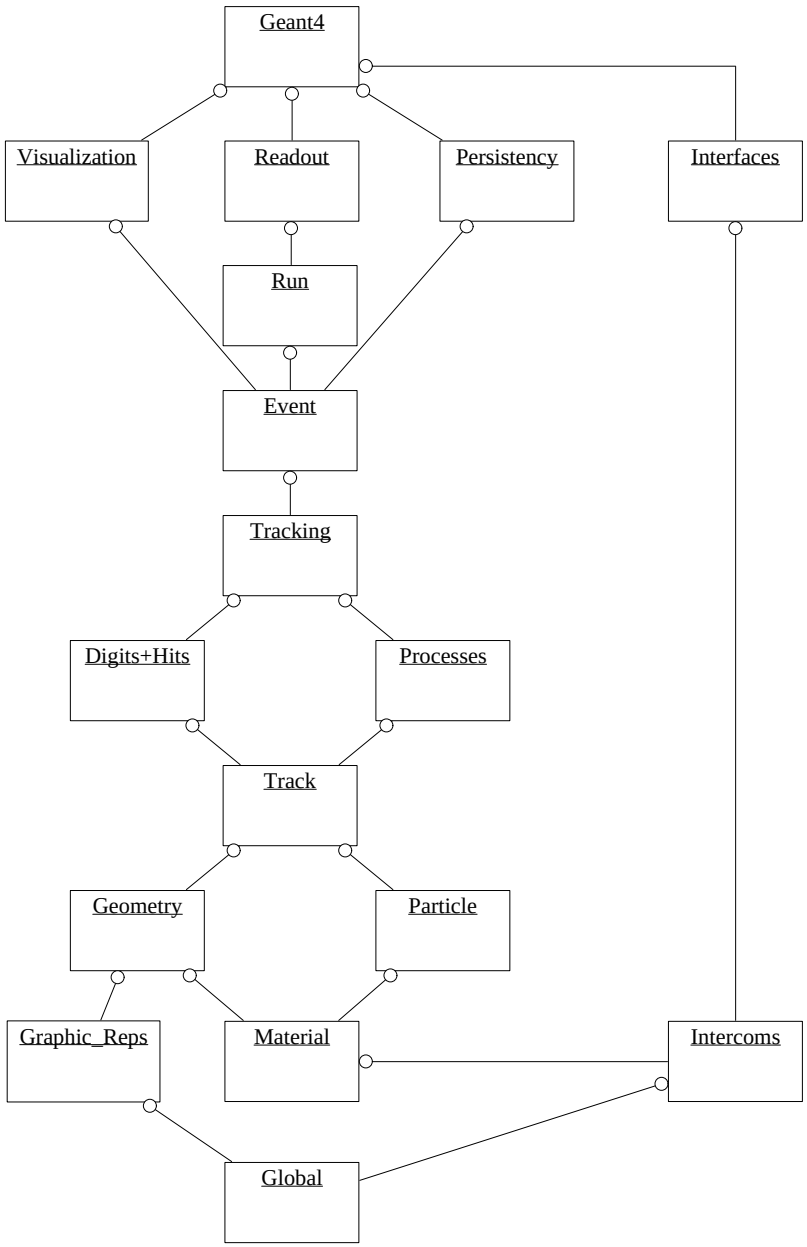


Figura 3.2: Schema dei domini che compongono Geant. Queste categorie sono collegate fra loro da una linea retta, il pallino al termine della linea indica la dipendenza dei domini (ad esempio la categoria *Tracking* è sottodominio di *Event* che a sua volta dipende dalle sopracclassi *Visualisation*,*Run* e *Persistency*).

- **Run e Event** permettono di generare l'evento primario e produrre le particelle secondarie che nascono dalle interazioni con il materiale attraversato.
- **Tracking e Track** sono legate alla propagazione di una particella, limitano la lunghezza dello step e gestiscono i processi fisici più importanti.
- **Hits e Digitization** gestiscono la parte sensibile del rivelatore, creano la sua risposta ad una particella che lo attraversa, generando un segnale logico e riproducendo la fase di digitalizzazione.
- **Visualization** determina la possibilità di visualizzare, attraverso librerie grafiche e sistemi CAD, la geometria che è stata definita e le tracce delle particelle generate e di quelle secondarie prodotte dalle interazioni. Gestisce i driver di visualizzazione di sistemi grafici quali OpenGL, OpenInventor (per Windows), DAWN, PostScript e VRML.

3.2.1 Geometry

La definizione di un rivelatore consiste nella rappresentazione dei suoi elementi geometrici, posizionati in un VolumeMadre, e dei materiali di cui sono costituiti. Geant richiede di definire per ogni elemento tre tipi di volumi: quello 'solido', che indica il tipo di figura geometrica che si considera e le dimensioni che la caratterizzano, quello 'logico', in cui si determina il materiale, e infine il volume fisico, che stabilisce la posizione dell'elemento geometrico nello spazio, il volume in cui è contenuto, le operazioni che vengono effettuate (traslazioni, rotazioni, operazioni booleane,...) e che assegna all'elemento un numero indicativo (copyNumber).

Figura 3.3: Rappresentazione grafica 3D di volumi definiti in un programma di simulazione attraverso le classi di Geant4.

Volume Solido

I solidi definiti nelle librerie di Geant sono quelli rappresentabili dallo standard di STEP. I più comuni sono i solidi CSG, *Constructive Solid Geometry*, descritti da un numero minimo di parametri necessari, per definirne le dimensioni.
Elenchiamo ora i principali costruttori delle forme geometriche più utilizzate:

1. **G4Box** definisce cubi e rettangoloidi di determinata larghezza sulle x, lunghezza sulle y e spessore sulle z tramite il costruttore:

   G4Box (const G4String nome,
   G4double semilarghezzaX,
   G4double semilunghezzaY,
   G4double semispessoreZ)

2. **G4Tubs** permette di costruire forme cilindriche, sezioni di cilindri e corone circolari, di raggio r, altezza h e apertura angolare θ:

   G4Tubs (const G4String nome,
   G4double raggioMinimo,
   G4double raggioMassimo,
   G4double semialtezzaZ,
   G4double angoloMinimo,
   G4double angoloMassimo)

3. **G4Cons** definisce coni e sezioni coniche attraverso un raggio r_1 della base conica, un raggio r_2 della cima, un'altezza h sulle z e un'apertura angolare θ:

   G4Cons (const G4String nome,
   G4double $r_1 Minimo$*,*
   G4double $r_1 Massimo$*,*
   G4double $r_2 Minimo$*,*
   G4double $r_2 Massimo$*,*
   G4double semialtezzaZ,
   G4double angoloMinimo,
   G4double angoloMassimo)

4. **G4Sphere** permette di costruire sfere, semisfere e corone sferiche, di raggio r, apertura θ e angolo azimutale ϕ:

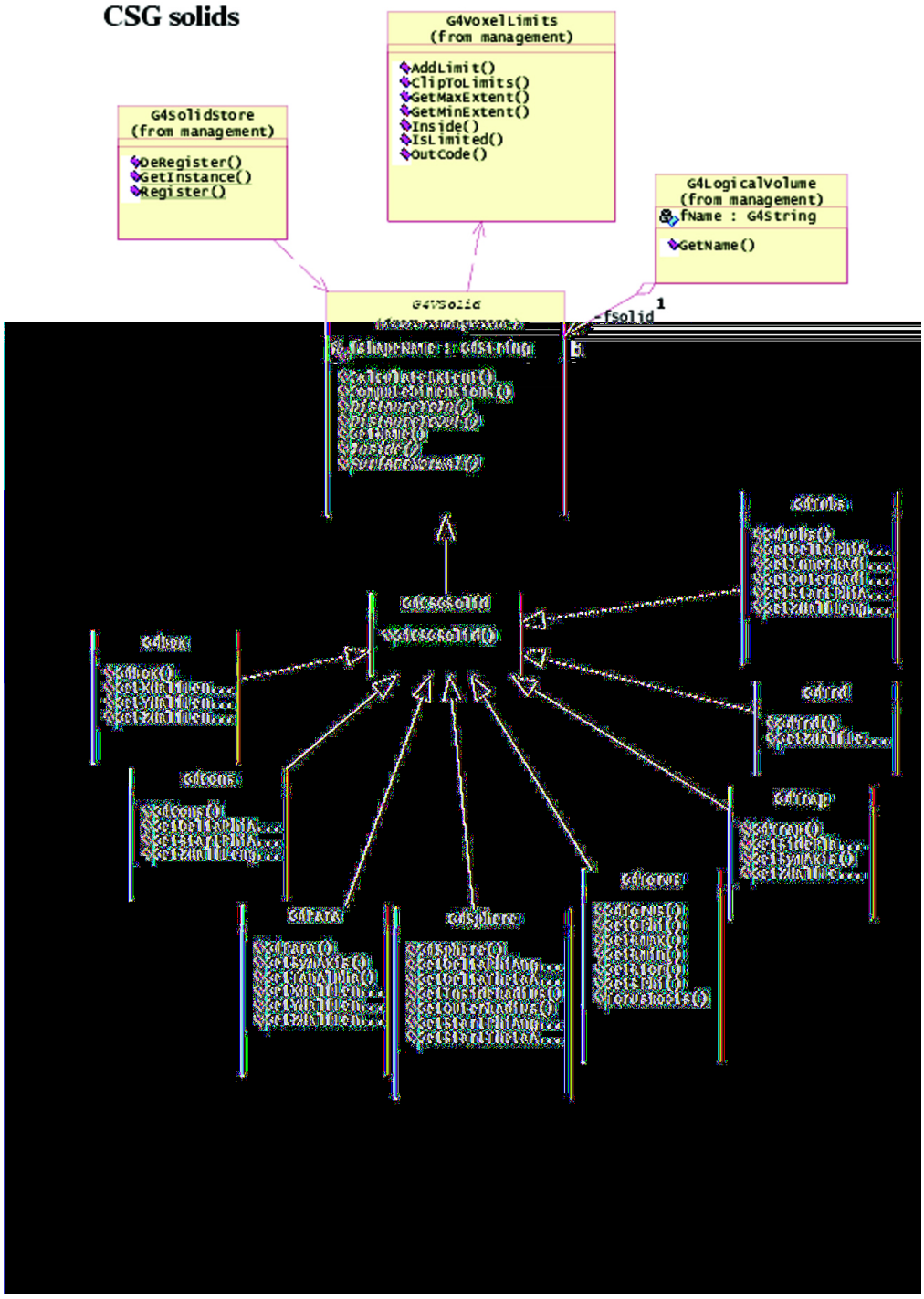


Figura 3.4: Diagramma UML, Unified Modelling Language, delle classi del dominio G4CSGSolid e dei suoi sottodomini.

G4Sphere (const G4String nome,
G4double rMinimo,
G4double rMassimo,
G4double θMinimo,
G4double θMassimo,
G4double φMinimo,
G4double φMassimo)

A questi si aggiungono costruttori per le forme toroidali (*G4Torus*), per i trapezoidi (*G4Trd* o *G4Trap*) e i parallelepipedi (*G4Para*).

E' possibile costruire forme geometriche più complesse relazionando due solidi di base, come quelli descritti nella categoria CSG, attraverso le operazioni booleane di unione, intersezione o sottrazione, traslando o ruotando la seconda figura rispetto alla prima. Le classi che possiamo utilizzare sono *G4UnionSolid*, *G4IntersectionSolid* e *G4SubtractionSolid*, i cui costruttori richiedono come parametri il nome e i due solidi interessati.
L'esempio seguente mostra l'unione di un cubo e un cilindro, centrati entrambi nell'origine. Il solido creato sarà un'istanza della classe G4UnionSolid, le sue caratteristiche fisiche verranno definite nel LogicVolume e nel PhysicVolume.

```
// definisce le variabili che determinano le dimensioni dei solidi
G4double sizeX = 10*cm;
G4double sizeY = 10*cm;
G4double sizeZ = 10*cm;
G4double raggio = 3*cm;

// definisce i due solidi
G4Box* cubo = new G4Box(``cubo'', sizeX/2, sizeY/2, sizeZ/2);
G4Tubs* cil = new G4Tubs(``cilindro'', 0, raggio, sizeZ/2, 0, 2*π);

// definisce il solido booleano dato dalla unione dei due precedenti
G4UnionSolid* union = new G4UnionSolid(``solUnion'', cubo, cil);
```

Questa implementazione permette di ottenere la configurazione della fig. 3.5,

dentro un WorldVolume definito invisibile per rendere più evidenti gli oggetti descritti.

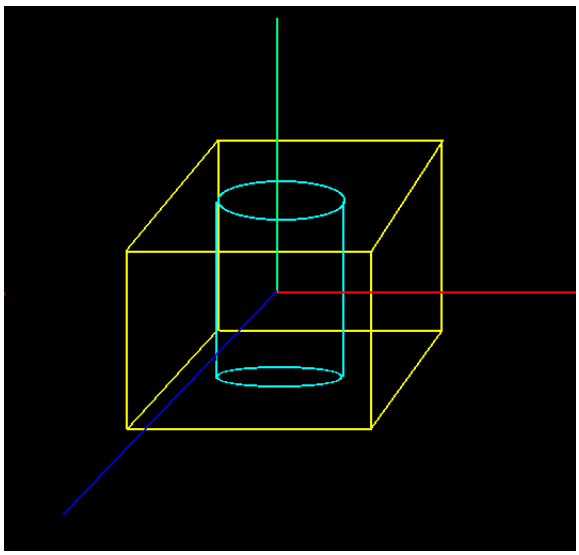


Figura 3.5: Rappresentazione grafica dell'unione di un cubo, di lato 10 *cm*, e un cilindro, di raggio 3 *cm*, centrati nell'origine degli assi.

Per effettuare sui solidi delle trasformazioni, traslazioni o rotazioni, si può usare rispettivamente le classi *G4ThreeVector* e *G4RotationMatrix*, definendo delle istanze di queste classi e utilizzando costruttori che le accettino come parametri. L'esempio seguente mostra l'unione di un cilindro e una semisfera, implementazione presente nella simulazione della camera Farmer[1]. La semisfera viene traslata di $h/2$ sull'asse del cilindro rispetto all'origine, l'intera figura ottenuta è poi ruotata di 90^o rispetto all'asse delle ascisse. Il solido costruito è un'istanza della classe G4UnionSolid, il materiale e le sue proprietà fisiche verranno definite dal logicVolume e dal physicVolume.

```
// definisce l'altezza del cilindro e il raggio dei due solidi
G4double sizeZ = 10*cm;
G4double raggio = 3*cm;
G4double phi = 90*deg;
```

[1]Farmer: è una camera a ionizzazione cilindrica, utilizzata per le misure di dose in ambito ospedaliero.

```
// definisce i due solidi
G4Tubs* cil = new G4Tubs(``cilindro'', 0, raggio, sizeZ/2, 0, 2*π);
G4Sphere* sfera = new G4Sphere(``sfera'', 0, raggio, 0, 2*π, 0, ϕ);

// definisce le variabili di traslazione e rotazione
G4ThreeVector trasl(0, 0, sizeZ/2);
G4RotationMatrix* rot;
rot.rotateX(ϕ);

// definisce il solido booleano dato dalla unione dei due precedenti
G4UnionSolid* newUnion =
new G4UnionSolid(``solNewUnion'', cil, sfera, rot, trasl);
```

La configurazione ottenuta è rappresentata nella fig. 3.6. Il cilindro è centrato nell'origine degli assi, la semisfera invece è traslata di metà dell'altezza del cilindro. La figura nasce attorno all'asse z, ma viene poi ruotata di 90^o rispetto ad x, sul piano yz.

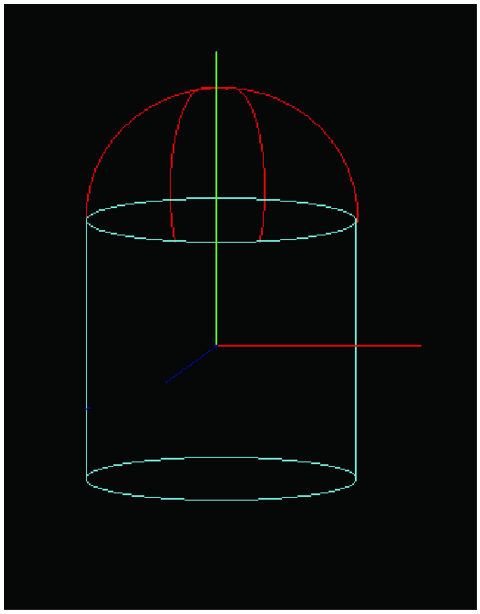


Figura 3.6: Rappresentazione grafica dell'unione di un cilindro e una semisfera.

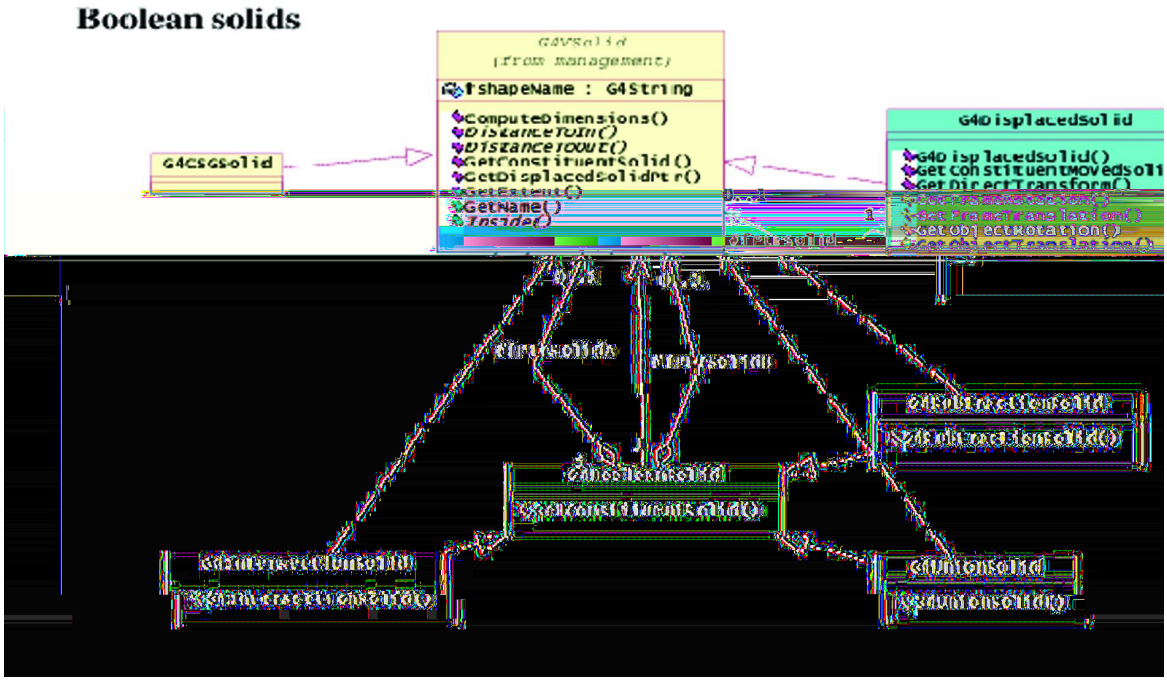


Figura 3.7: Diagramma UML delle classi riguardanti le operazioni booleane fra solidi.

Volume Logico

Il volume logico gestisce le informazioni associate agli elementi del rivelatore, come il materiale di cui è costituito, il colore con cui lo si vuole visualizzare, i campi che lo attraversano, etc ... , trascurando le proprietà fisiche relative alla sua localizzazione nello spazio.
Elenchiamo i due costruttori che è possibile utilizzare per definire un volume logico:

- *G4LogicalVolume (G4VSolid* solido,*
  G4Material materiale,*
  const G4String nome)

- *G4LogicalVolume(G4VSolid* solido,*
  G4Material materiale,*
  const G4String nome,
  G4FieldManager campi,*
  G4VSensitiveDetector parteSensibile,*
  G4UserLimits cuts)*

Attraverso il volume logico possiamo determinare un colore per la visualizzazione grafica dell'elemento geometrico definito, il colore di default è bianco su sfondo nero. Riportiamo in seguito come esempio l'implementazione del LogicVolume del solido 'union' creato nella sezione precedente e del VolumeMadre che lo contiene.

```
G4Material* defaultMaterial = Air;
G4Material* solidMaterial = Water;

// definizione del VolumeMadre e dichiarazione di invisibilità
G4LogicalVolume* logicWorld =
new G4LogicalVolume(solidWorld, defaultMaterial, ‘‘World’’);
logicWorld -> SetVisAttributes(G4VisAttributes::Invisible);

// definizione del VolumeFiglio che vogliamo colorato di azzurro
G4LogicalVolume* logicUnion =
```

```
new G4LogicalVolume(union, solidMaterial, ‘‘logUnion’’);
G4VisAttributes* AttrUnion = new G4VisAttributes(G4Colour(0.5,3.0.1.0));
logicUnion -> SetVisAttributes(AttrUnion);
```

Il volume logico gestisce anche le informazioni relative alla gerarchia dei 'pacchetti' richiesti dalla parametrizzazione FastMonteCarlo.

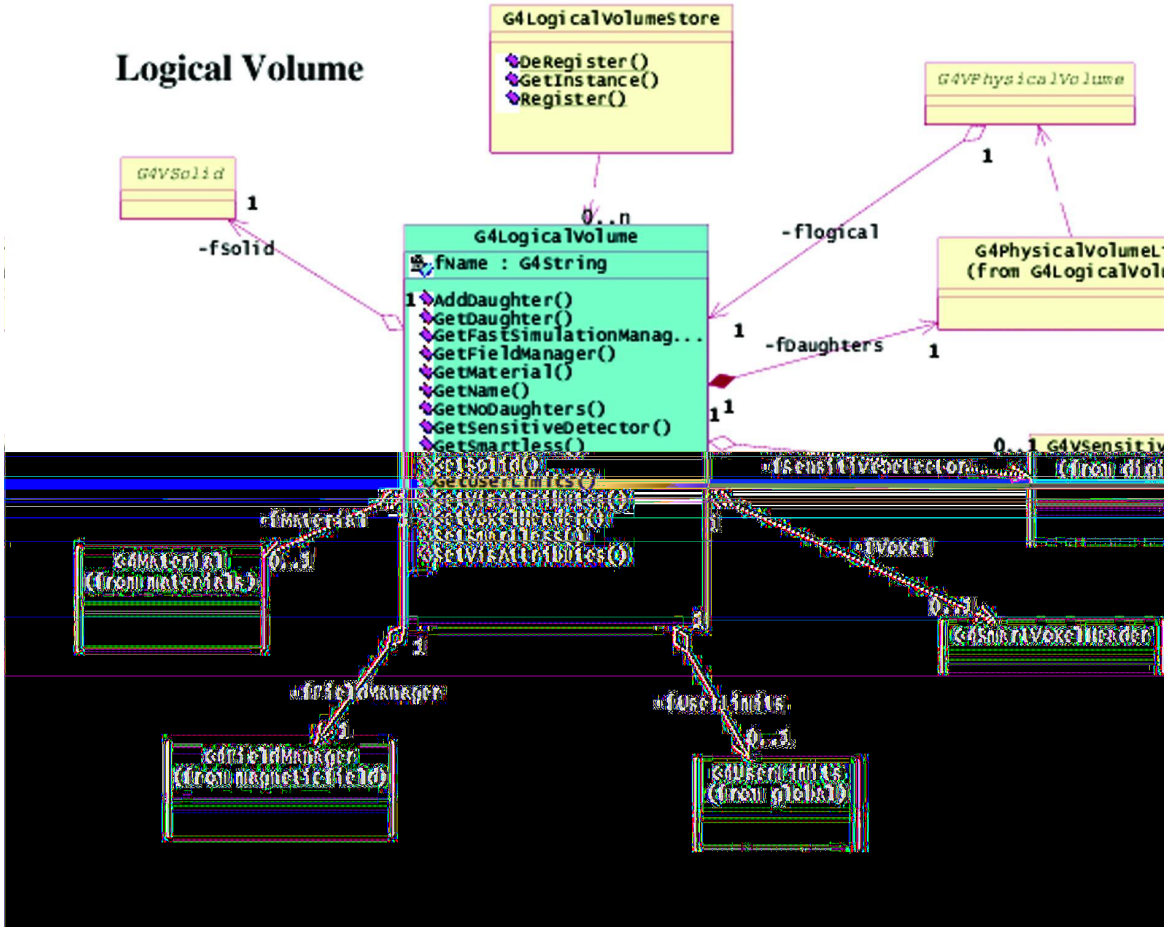


Figura 3.8: Diagramma UML della classe G4LogicalVolume e dei collegamenti di questa con le classi adiacenti.

Volume Fisico

Il volume fisico definisce la posizione spaziale degli elementi del rivelatore che sono stati definiti. Possono essere usate diverse tecniche, dalla dislocazione semplice di un elemento o di una coppia, al posizionamento ripetuto di un determinato volume, attraverso l'utilizzo di formule lineari semplici o di funzioni specifiche più complesse. Il posizionamento nello spazio di un volume utilizza una matrice di trasformazione, che, per il posizionamento ripetuto, è funzione anche del numero di volte e della distanza in una determinata direzione.

I costruttori definiti in Geant sono:

- *G4PVPlacement(G4RotationMatrix* rot,*
  const G4ThreeVector(traslX, traslY, traslZ),
  const G4String nome,
  G4LogicalVolume logVol,*
  G4VPhysicalVolume phyWorld,*
  G4bool valBool,
  G4int copyNb)

- *G4PVPlacement(G4Transform3D solidTransform,*
  const G4String nome,
  G4LogicalVolume logVol,*
  G4VPhysicalVolume phyWorl,*
  G4bool valBool,
  G4int copyNb)

Queste funzioni definiscono il tipo di trasformazione da applicare al solido (0 se non è necessaria nessuna rotazione, G4ThreeVector(0, 0, 0) se questo rimane centrato nell'origine), il volume logico a cui è associato e il VolumeMadre fisico in cui è contenuto. L'unico volume che non sta in uno più grande è il WorldVolume, in cui sono contenuti tutti gli altri elementi del rivelatore, il suo costruttore avrà dunque uno 0 nella voce riguardante il VolumeMadre. Queste funzioni richiedono anche come parametri una variabile logica (true/false), sulle possibili operazio-

ni booleane che verranno effettuate sul volume, e un copyNumber, indicativo dell'elemento.
Il metodo *Replicas* permette di ripetere molteplici copie identiche di un volume definito, attraverso il costruttore

- *G4PVReplica(const G4String nome,*
  G4LogicalVolume logVol,*
  G4PhysicalVolume phyVolMother,*
  const EAxis asse,
  const G4int nRepl,
  const G4double lunghezza,
  const G4double offset)

Questo metodo è uilizzato normalmente solo quando i volumi sono posizionati in una determinata simmetria lineare o rotazionale, lungo coordinate cartesiane o cilindriche, e sono descritti dai solidi CSG.

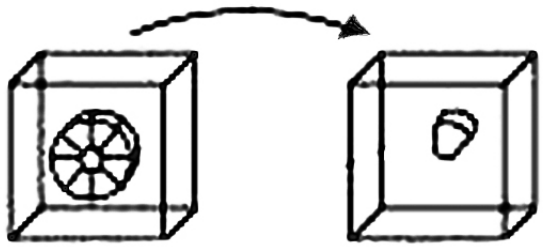


Figura 3.9: Esempio del risultato ottenuto applicando il metodo *Replicas* ad un volumetto elementare.

Nel caso in cui i volumi vengano ripetuti, ma abbiano differenti dimensioni o siano costituiti da diversi materiali, si può utilizzare il metodo *Parameterised-Volumes*. Tutte le caratteristiche del solido, anche la matrice di trasformazione, sono in funzione del copyNumber. L'utente implementa la funzione parametrica

desiderata, dopodichè il programma aggiorna automaticamente le informazioni associate al volume fisico.
Il costruttore generico è:

- *G4PVParameterised(const G4String nome,*
  G4LogicalVolume logVol,*
  G4PVPhysicalVolume phyMotherVol,*
  const EAxis asse,
  const G4int nRepl,
  G4VPVParameterisation param)*

L'istanza 'param' contenente le proprietà dei volumi è invece oggetto della classe *G4VPVParameterisation*, il costruttore deve essere definito dall'utente a seconda delle variabili di cui ha bisogno per l'implementazione del rivelatore. Si riporta come esempio l'utilizzo della parametrizzazione nella definizione di camere cubiche di dimensioni $(20x20x20)\ cm^3$, il cui volume solido sarà un'istanza di G4Box e il volume logico di G4LogicalVolume.

```
// definizione dei parametri
G4VPVParameterisation* param = new EsParameterisation(
G4int n,           // numero camere
G4double z,        // centro della prima
G4double d,        // spazio fra i centri
G4double p,        // profondita' delle camere
G4double l_i,      // lunghezza iniziale
G4double l_f);     // lunghezza finale

G4VPhysicalVolume* phyCam = new G4PVParameterised(
"phyCam,           // nome
 logCam,           // volume logico
 phyMotherVol,     // volume madre
 zAsse,            // asse
 n,                // numero di camere
 param);           // parametrizzazione
```

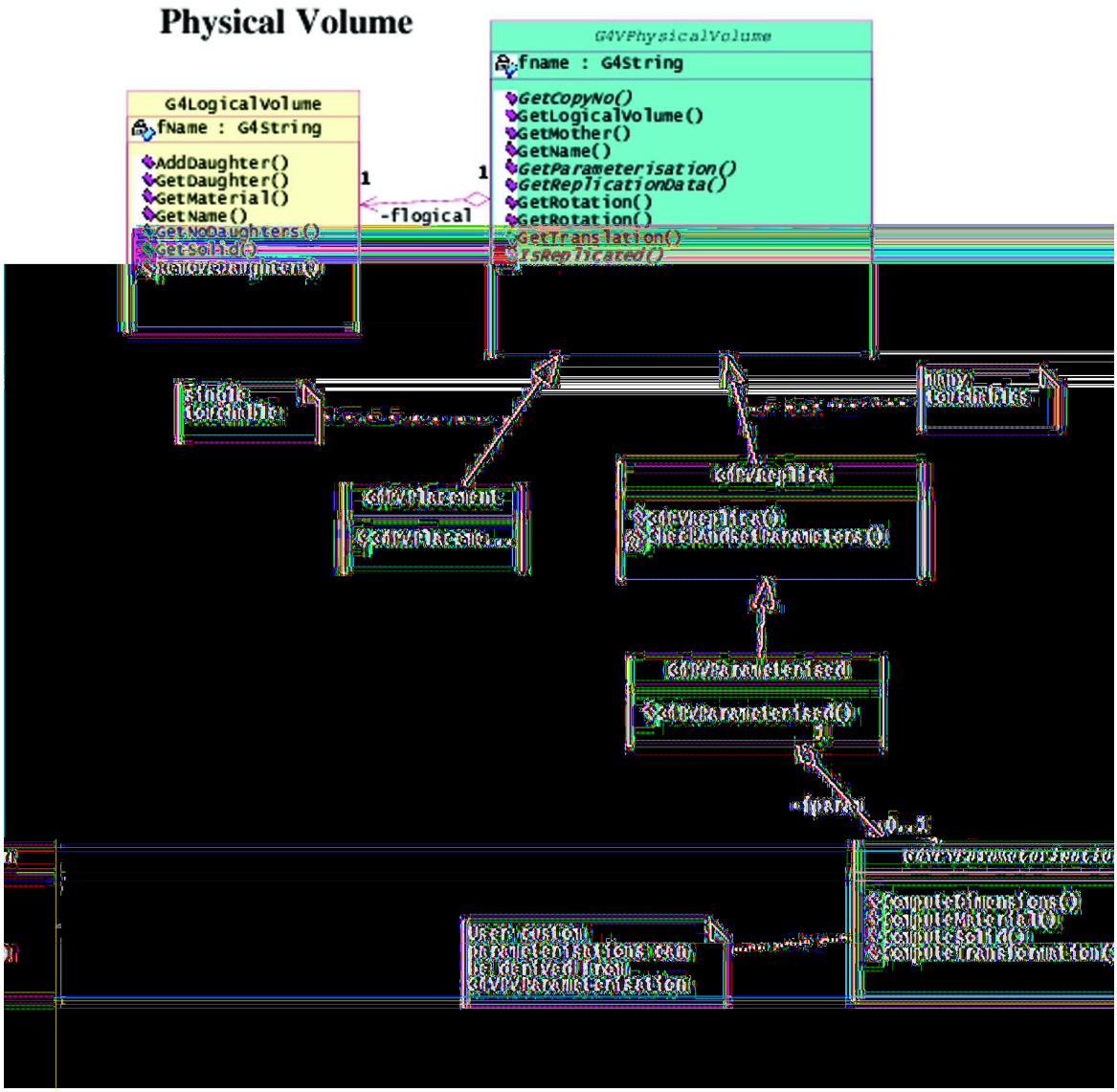


Figura 3.10: Diagramma UML della classe G4PhysicalVolume.

3.2.2 Materials

In natura i materiali, composti chimici o miscele, sono costituiti da elementi, e questi a loro volta da isotopi. Geant permette di definire ognuna di queste tre categorie attraverso le classi:

- *G4Material*, che descrive le proprietà macroscopiche del materiale, quali densità, temperatura, pressione, etc ... ,
- *G4Element*, che definisce le proprietà degli atomi, come l'effettivo numero atomico, l'effettiva massa per mole, l'energia della shell, etc ... ,
- *G4Isotope*, che caratterizza il tipo di atomo, attraverso il numero atomico, il numero di nucleoni, la massa per mole, etc ...

Ciascuna di queste classi contiene delle tabelle di dati utilizzati dalle istanze create e dai metodi definiti. L'unica classe visibile dal resto del toolkit di Geant è G4Material, utilizzata dai domini Geometry, Tracking e Physics, essa contiene tutte le informazioni relative agli elementi e agli isotopi di cui un materiale è costituito.

Isotopi

Un'istanza di questa classe accetta come parametri il nome, il numero atomico, il numero di nucleoni, la massa per mole e un numero indicativo che permette al costruttore di immagazzinare automaticamente l'oggetto così definito nella 'tavola degli isotopi'.
Il costruttore generico è:

- *G4Isotope(G4String nome,*
  G4int protonsNumber,
  G4int nucleonsNumber,
  G4double moleMassa)

Per definire un elemento costituito da isotopi, all'istanza della classe G4Element si applica il metodo *AddIsotope* così dichiarato:

- *AddIsotope(G4Isotope* isotopo,*
  G4double percentuale)

Viene riportato un esempio pratico dell'implementazione di due isotopi dell'Uranio e della definizione del materiale generato da questi.

```
// definizione di due isotopi dell'Uranio
G4double a5 = 235.01*g/mole;
G4double a8 = 238.03*g/mole;
G4Isotope* U5 = new G4Isotope(``U235'', 92, 235, a5)
G4Isotope* U8 = new G4Isotope(``U238'', 92, 238, a8)

// definizione dell'elemento Uranio
G4int compNb = 2;
G4Element* U = new G4Element(``uranio'', ``U'', compNb);
U->AddIsotope(U5, abundance = 0.9);
U->AddIsotope(U8, abundance = 10.*perCent);

// per stampare la tavola degli isotopi aggiornata
cout<< *(G4Isotope::GetIsotopeTable()) <<endl;
```

Elementi

Un'istanza della classe G4Element è caratterizzata da un nome, un simbolo, lo effettivo numero atomico e di nucleoni, l'effettiva massa per mole, il numero di isotopi e un indice di collocazione nella tavola degli elementi. All'oggetto così definito viene anche associato un vettore di puntatori agli isotopi che lo compongono e un vettore delle relative percentuali in cui questi sono presenti. Ogni volta che viene definito un elemento il programma aggiorna automaticamente la tabella in memoria.
Il costruttore di questa classe è:

- *G4Element(G4String nome,*
  G4String simbolo,
  G4double nProtoni,
  G4double moleMassa)

Segue un esempio di definizione dell'ossigeno e dell'idrogeno, attraverso le variabili z, numero medio di protoni, e a, massa di una mole:

```
G4double z = 8.0;
G4double a = 16.00*g/mole;
G4Element* O = new G4Element(``Oxygen'', ``O'', z, a);

G4double z = 1.0;
G4double a = 1.01*g/mole;
G4Element* H = new G4Element(``Hydrogen'', ``H'', z, a);

// stampa della tavola degli elementi aggiornata
cout << *(G4Element::GetElementTable()) << endl;
```

Materiali

Un materiale è definito da un nome, una densità , uno stato fisico (solido per default), temperatura e pressione, i cui valori di default sono 25^o e 1 atm. Viene associato al materiale anche il numero di elementi, un vettore di puntatori ai vari elementi, un vettore contenente la quantità presente nel materiale di ciascun elemento, un vettore del numero di atomi di ogni elemento e infine l'indice per inserire il materiale nella tabella.

Il costruttore della classe è:

- *G4Material(G4String nome,*
  G4double nProtoni,
  G4double moleMassa,
  G4double densità)

Come esempio definiamo alcuni materiali con differenti metodi:

```
// attraverso il costruttore
G4double density = 2.700 * g/cm^3;
G4double a = 26.98 * g/mole;
G4double z = 13.0;
G4Material* Al = new G4Material(``Aluminum'', z, a, density);
```

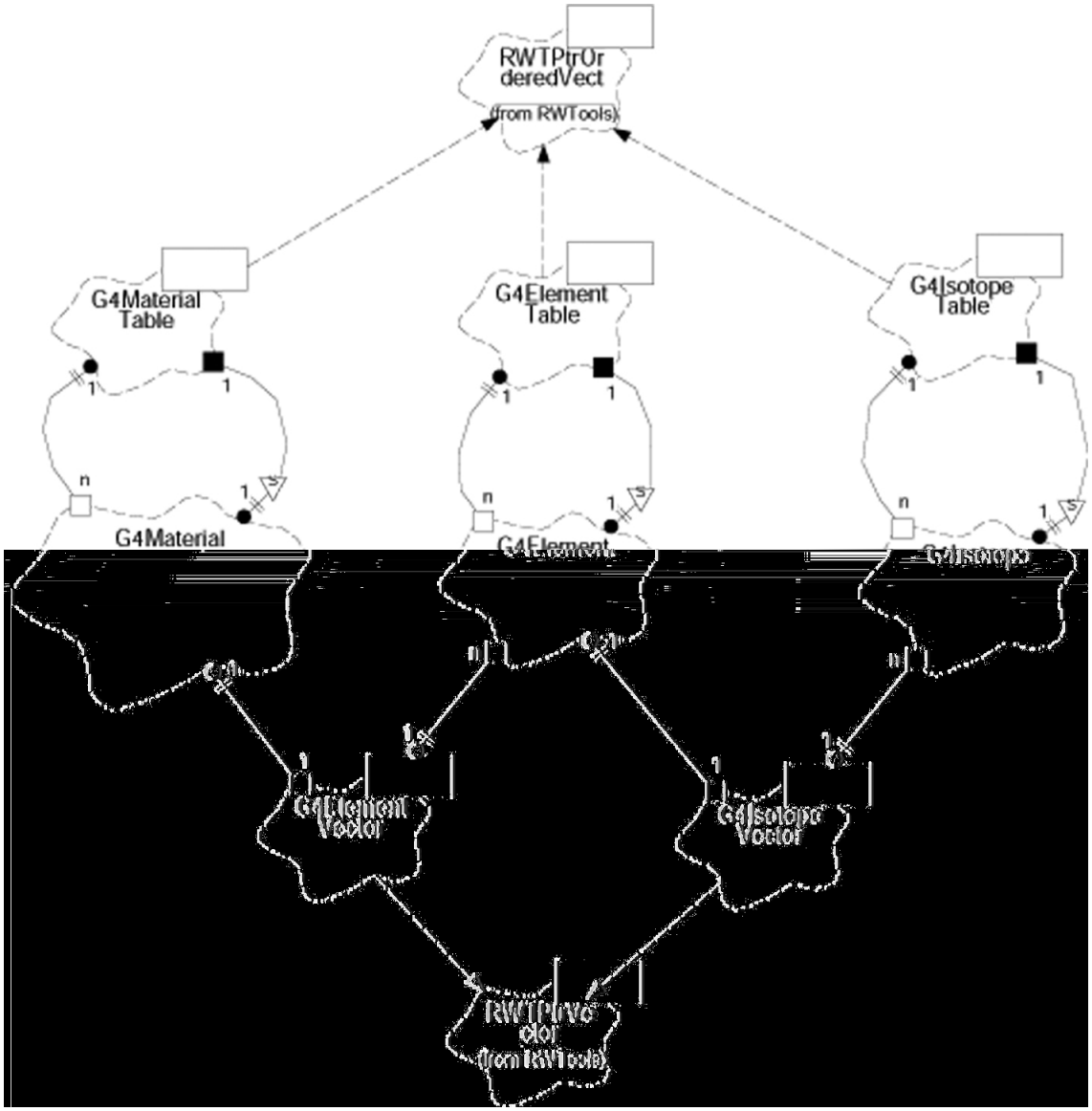


Figura 3.11: Correlazione fra le classi appartenenti al dominio Materials.

```
// attraverso gli elementi di cui è composto
// 1^o CASO : molecola chimica
G4double density = 1.000 * g/cm^3;
G4int nComponents = 2;
G4Material* H2O = new G4Material(``Water'', density, nComponents);
H2O->AddElement(H, natoms = 2);
H2O->AddElement(O, natoms = 1);

// 2^o CASO : miscela di componenti
G4double density = 1.290 * g/cm^3;
G4int nComponents = 2;
G4Material* Air = new G4Material(``Air'', density, nComponents);
Air->AddElement(N, fractionmass = 0.7);
Air->AddElement(O, fractionmass = 0.3);

// 3^o CASO : miscela di miscele
G4double density = 0.200 * g/cm^3;
G4int nComponents = 3;
G4Material* Aerog = new G4Material(``Aerogel'', density, nComponents);
Aerog->AddMaterial(SiO2, fractionmass = 62.5*perCent);
Aerog->AddMaterial(H2O, fractionmass = 37.4*perCent);
Aerog->AddElement(C, fractionmass = 0.1*perCent);

// per stampare un solo materiale o l'intera tabella
G4cout << Air << endl;
G4cout << *(G4Material::GetMaterialTable()) <<endl;
```

La classe automaticamente verifica per ogni materiale se il totale delle percentuali (fractionsmass) degli elementi che lo compongono è corretta, altrimenti avvisa l'utente che potrebbero esserci errori nei risultati. Ad ogni istanza della classe viene assegnato un indice utile per il calcolo delle sezioni d'urto. Questo avviene infatti attraverso l'utilizzo di una tabella che ha i materiali sulle righe e l'energia della particella incidente sulle colonne. Se non viene definito un indice, il programma cerca il materiale attraverso dei metodi sequenziali.

3.2.3 Run

In Geant il run è il dominio più vasto, esso consiste in una sequenza di eventi. Durante un run la geometria del rivelatore, l'impostazione del volume sensibile e processi fisici utilizzati non devono essere modificati. Il run inizia attraverso la chiamata del metodo beamOn() applicato ad un oggetto della classe *G4RunManager*. Ogni run ha un numero identificativo definito dall'utente, che però non è utilizzato dal kernel di Geant.
I metodi più importanti di questa classe sono:

- *initialize()*: inizializzazione della geometria del rivelatore e del volume sensibile, delle particelle e dei processi fisici che vengono considerati, delle tabelle utilizzate nel calcolo delle sezioni d'urto. Questo metodo deve essere chiamato precedentemente al primo run e verrà invocato automaticamente per i run successivi nel caso in cui alcune quantità vengano modificate.

- *beamOn(G4int *)*: questo metodo gestisce la simulazione di un run. L'intero che viene passato come parametro indica il numero di eventi che devono essere simulati.

- *abortRun()*: permette di interrompere un run in corso.

Una classe di questo dominio utile all'utente è *G4UserRunAction*, in quanto contiene due metodi virtuali utilizzati in ogni simulazione:

- *beginOfRunAction()*: metodo chiamato prima di `beamOn()`. Assegna un numero identificativo al run, permette di creare un istogramma e definisce le condizioni al contorno della parte sensibile del rivelatore e dei moduli di digitalizzazione.

- *endOfRunAction()*: questo metodo è utilizzato solo al termine del run. Ha il compito di riempire e stampare l'istogramma e gestire il sommario del run, in cui vengono riportati il tempo di durata della simulazione, il numero di eventi simulati e gli indici random utilizzati.

3.2.4 Event

La classe *G4Event* permette di generare, seguire e terminare un evento. Un oggetto di questa classe contiene tutti gli imput e gli output dell'evento simulato,

fra cui il vertice e il tipo di particella inviata, le traiettorie seguite, gli urti avvenuti nella parte sensibile del rivelatore e gli impulsi generati.
Due metodi virtuali vengono utilizzati dall'utente:

- *beginOfEventAction()*: questo metodo è utilizzato prima di convertire la particella primaria in un'oggetto della classe *G4Track*. Esso inizializza e registra gli istogrammi di ogni particolare evento.
- *endOfEventAction()*: metodo che viene chiamato solo alla fine dell'evento. Esso viene utilizzato per l'analisi dell'evento appena simulato.

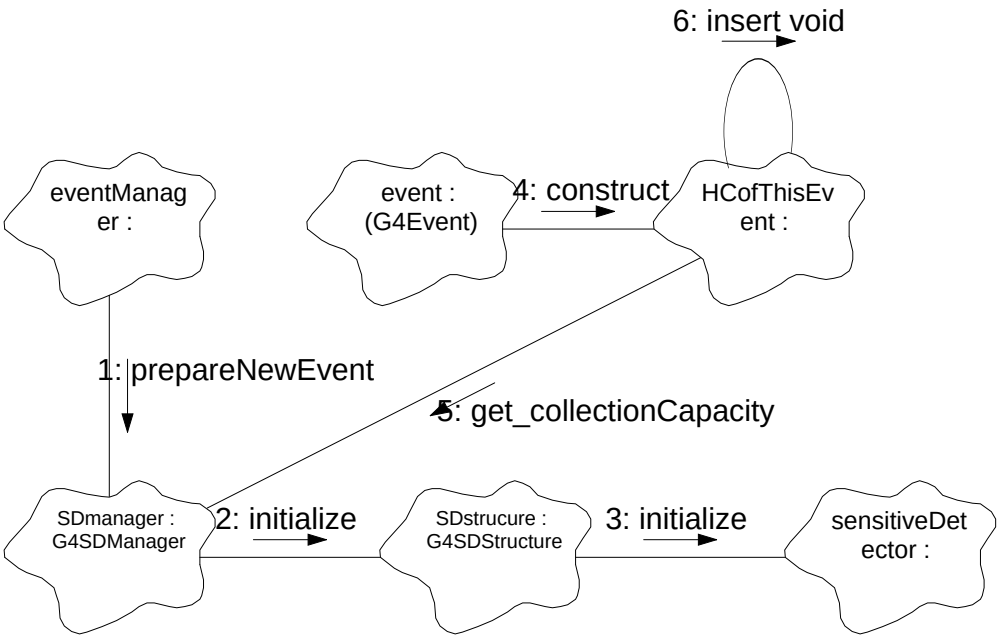


Figura 3.12: Schema delle classi in gioco all'inizio dell'evento

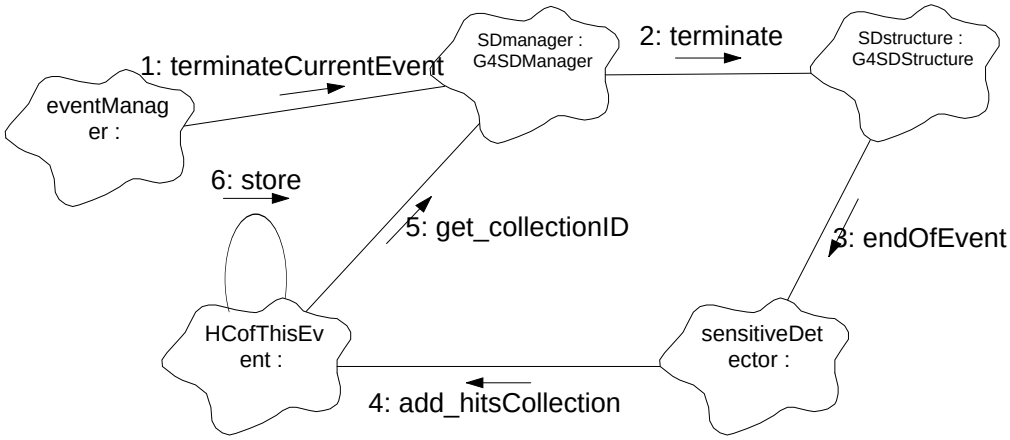


Figura 3.13: Schema delle classi che intervengono al termine dell'evento.

3.2.5 Particle

Esistono tre principali classi che permettono di definire completamente una particella in Geant4: *G4ParticleDefinition* permette di determinare il tipo di particella, attraverso il nome, la massa, lo spin, la vita media e il tipo di decadimento, *G4DynamicParticle* descrive le interazioni di questa con il materiale che attraversa, con informazioni sull'energia, sul momento e sulla polarizzazione, e infine *G4Track* segue i suoi spostamenti nel tempo e nello spazio, considerando la traiettoria nel rivelatore, il tempo, la posizione e gli step stabiliti. Quasi tutte le particelle fino ad oggi conosciute sono descritte in Geant4.
Vengono divise in tre gruppi:

- *particelle elementari*: tutte le particelle che percorrono un tratto di lunghezza finita e che interagiscono con il materiale del rivelatore. Sono incluse, però, anche alcune particelle con una vita media piccola. Le particelle di questo gruppo vengono ancora suddivise in **stabili**, quelle che non decadono o hanno piccola probabilità di decadere (protoni, elettroni, neutroni e fotoni); **di lunga vita media**, superiore a 10^{-14} *sec* (muoni e pioni ca-

richi); **di corta vita media**, nel caso di K^0 che decade in K_S^0 o K_L^0; **di decadimento immediato**, come π^0 e η. A queste si aggiungono due classi particolari: **gli optical photons**, usati in alcuni casi specifici, come nell'effetto Čerenkov e nella luce di scintillazione, e **il geantino**, una particella virtuale utilizzata nelle simulazioni in cui non si considerano le interazioni con i materiali, ma si studia solo il trasporto di energia.

- *nuclei*: solo alcuni tipi di nuclei possono essere utilizzati in Geant4, come le particelle α, nuclei dell'U^{238} e stati eccitati del C^{14}. Vengono divisi in due gruppi in base al tipo di implementazione: **nuclei leggeri**, usati di frequente nelle simulazioni, tra i quali le particelle α (He^4), il deuterio, l'elio He^3 e il tritone; **nuclei pesanti**, tutti i nuclei non descritti nella prima categoria.

L'utente è tenuto a definire i parametri di cut-off in range e i processi fisici che devono essere considerati. Deve quindi creare una classe derivata da *G4VUserPhisicsList* e implementare i metodi `ConstructParticle()`, `SetCuts()` e `ConstructProcess()`, quest'ultima trattata nella sezione seguente.
Il costruttore viene definito dalle seguenti istruzioni:

```
SimulatioPhysicsList::SimulatioPhysicsList():G4VUserPhysicsList()
{
// si possono definire i valori di default
defaultCutValues = 2.0*mm;
}
```

Viene in seguito riportato un esempio per specificare una o più particelle tramite il metodo `ConstructParticle()`.

// definizione di un protone e un elettrone

```
void SimulatioPhysicsList::ConstructParticle()
{
G4Proton::ProtonDefinition();
G4Electron::ElectronDefinition();
}
```

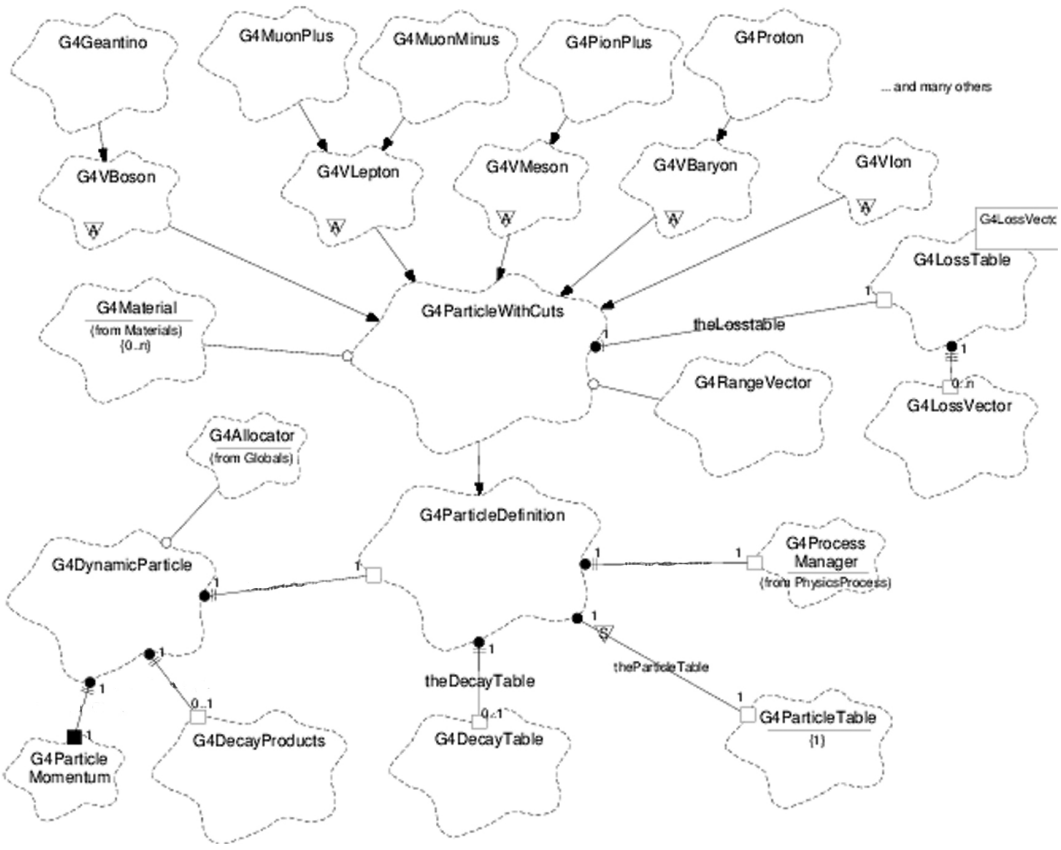


Figura 3.14: Schema del dominio Particle e dei suoi sottodominii

Se il numero di particelle da specificare è alto si possono definire intere classi di particelle (G4LeptonConstructor, G4MesonConstructor, G4BarionConstructor, G4BosonConstructor, G4IonConstructor e G4ShortlivedConstructor).
Ad esempio per attivare la classe dei leptoni basta il codice:

```
void SimulatioPhysicsList::ConstructLepton()
{
G4LeptonConstructor lep;
lep.ConstructParticle();
}
```

Il metodo `SetCuts()` definisce il valore di taglio in range di ogni particella. Questa viene dunque tracciata fino a quando la sua energia non raggiunge il valore per il quale il suo range residuo è pari al valore di taglio. L'energia rimanente viene allora rilasciata localmente nel punto in cui si ferma la particella. Può anche essere definito un unico *cut in range* valido per tutte le particelle, il suo valore di default è a 1 mm.
Viene definito attraverso le seguenti istruzioni:

```
void SimulatioPhysicsList::SetCuts()
{
SetCutsWithDefault();
}
```

Se si vogliono fissare valori di taglio differenti, a seconda del tipo di particella, è necessario seguire un certo ordine, partendo dai fotoni, dagli elettroni e dai positroni, per passare poi ai protoni e antiprotoni. Si può a questo proposito utilizzare il metodo `SetCutValue(G4double cut, G4String particle)` che associa ad ogni particella il suo *cut in range*. Sono definiti altri due metodi per la definizione del valore di taglio: `SetCutValueForOthers(G4double cut)`, permette di definire il *cut* per tutte le particelle restanti, `SetCutValueForOthersThan(G4double cut, G4particleDefinition* particle)`, permette di definirlo per tutte le restanti tranne una.
I *cut in range* sono trasformati dal programma in *cut in energy* a seconda del materiale e della particella.

3.2.6 Physics Processes

La categoria riguardante i processi fisici ha il compito di descrivere le interazioni possibili fra la particella e il materiale. Queste vengono divise in elettromagnetiche, adroniche, di trasporto di energia, di decadimento, ottiche, fotoleptoniche e di parametrizzazione. Tutti i processi fanno riferimento alla classe *G4VProcess*, ogni istanza di questa è definita da un nome e da un processo_tipo di identificazione.

Le interazioni elettromagnetiche implementate in Geant4, secondo la fisica descritta nel primo capitolo, vengono divise in:

Standard

1. Fotoni:
    - *G4ComptonScattering* implementa l'effetto Compton;
    - *G4GammaConversion* per la produzione di coppie;
    - *G4PhotoElectricEffect* riguardante l'effetto fotoelettrico;
2. Elettroni e Positroni:
    - *G4eBremsstrahlung o G4IeBremsstrahlung* implementa la perdita di energia per Bremsstrahlung;
    - *G4eIonisation o G4IeIonisation* per la perdita di energia per ionizzazione;
    - *G4eplusAnnihilation o G4IeplusAnnihilation* per l'annichilazione e^-/e^+;
    - *G4eEnergyLoss o G4IeEnergyLoss* descrivono la perdita continua di energia derivante dai primi due effetti combinati fra loro;
    - *G4SynchrotronRadiation* riguardante la radiazione di sincrotrone;
3. Adroni:
    - *G4hIonisation* per la perdita di energia per ionizzazione;
    - *G4hEnergyLoss* descrivono la perdita continua di energia;

Low Energy

1. Fotoni:
    - *G4LowEnergyCompton* implementa l'effetto Compton;
    - *G4LowEnergyRayleigh* per descrivere la diffusione Rayleigh;
    - *G4LowEnergyGammaConversion* per la produzione di coppie;
    - *G4LowEnergyPhotoElectric* riguardante l'effetto fotoelettrico;
2. Elettroni e Positroni:
    - *G4LowEnergyBremsstrahlung* implementa la perditas di energia per Bremsstrahlung;
    - *G4LowEnergyIonisation* per la perdita di energia per ionizzazione;
3. Adroni:
    - *G4hLowEnergyIonisation* per la perdita di energia per ionizzazione;

Esiste anche la classe (*G4MultipleScattering o G4IMultipleScattering*), utilizzata per descrivere lo scattering multiplo di tutte le particelle cariche. Per gli elettroni e lo scattering multiplo, Geant implementa ogni interazione in due modi differenti, con approccio differenziale o integrale, per questo troviamo sempre due classi, che determinano due modi diversi di calcolare l'energia depositata nel materiale.
Le interazioni radiazioni-materia che riguardano i raggi X sono trattate dalle classi *G4Cerenkov* e *G4TransitionRadiation*, nelle quali sono implementati l'effetto Čerenkov e la radiazione di transizione.
Segue un esempio di chiamata di alcuni processi:
Si definiscono le classi dei processi da utilizzare

```
void SimulatioPhysicsList::ConstructProcess()
{
AddTransportation(); // per implementare il trasporto di energia
ConstructEM();       // per implementare i construtti elettromagnetici
ConstructGeneral();  // per definire costrutti generali (ex. il decadimento)
}
```

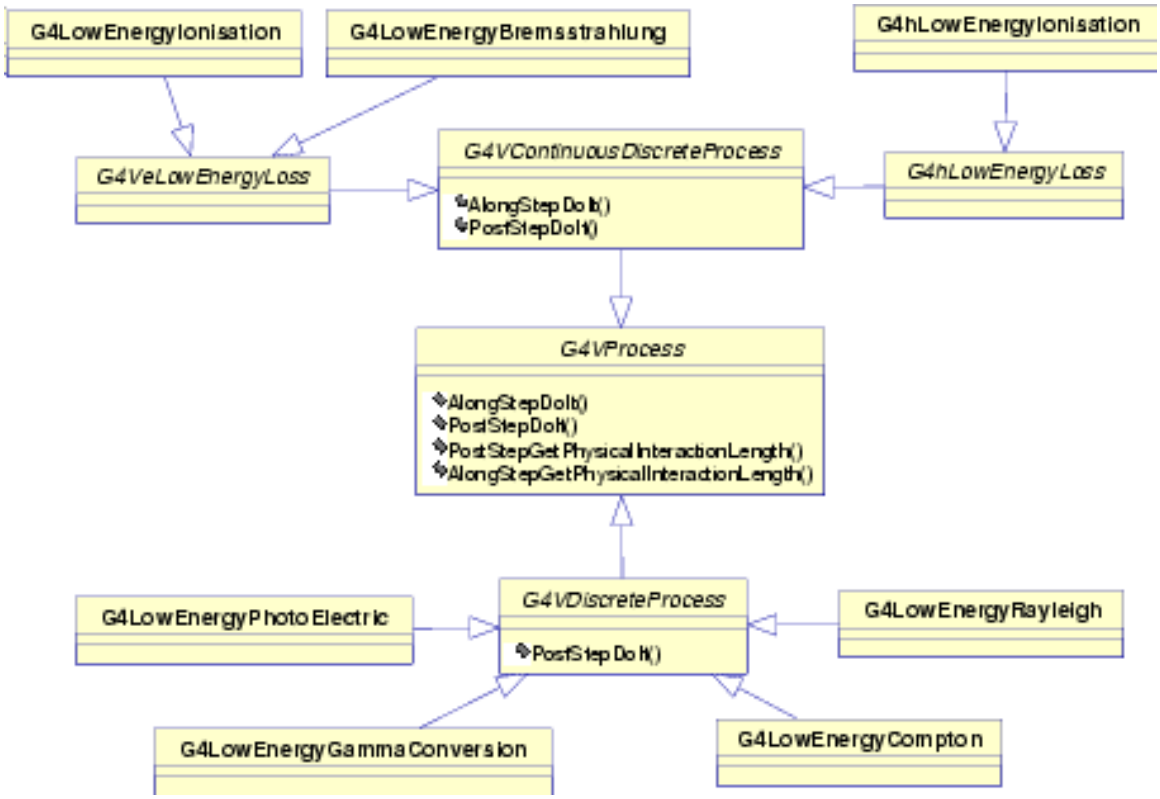


Figura 3.15: Schema dei processi a bassa energia implementati in Geant.

Processi elettromagnetici

```
void SimulatioPhysicsList::ConstructEM()
{
// si creano le istanze
G4ParticleDefinition* particle = G4Gamma::GammaDefinition;
G4ProcessManager* proc = particle->GetProcessmanager();
G4String partName = particle->GetParticleName();

// si creano i processi dei fotoni
thePhotoElettricEffect = new G4PhotoElectricEffect();
theComptonScattering = new G4ComptonScattering();
theGammaConversion = new G4GammaConversion();

// vengono aggiunti i processi creati
// all'istanza di G4ProcessManager
proc->AddDiscreteProcess(thePhotoElectricEffect);
proc->AddDiscreteProcess(theComptonScattering);
proc->AddDiscreteprocess(theGammaConvertion);
}
```

Decadimenti

```
#include "G4Decay.hh"

void CameraPhysicsList::ConstructGeneral()
{
// si creono le istanze
G4Decay* theDecayProcess = new G4Decay();
G4ParticleDefinition* particle = theParticleIterator->value();
G4ProcessManager* pmanager = particle->GetProcessManager();

// si verifica se il decadimento
// è possibile per la particella in questione
if (theDecayProcess->IsApplicable(*particle))
{
```

```
// si aggiunge il processo
pmanager ->AddProcess(theDecayProcess);

// vengono fissati gli indici
// che definiscono il tipo di decadimento
pmanager ->SetProcessOrdering(theDecayProcess, idxPostStep);
pmanager ->SetProcessOrdering(theDecayProcess, idxAtRest);

}

}
```

Figura 3.16: Schema delle classi implementate in Geant4 per i processi di decadimento.

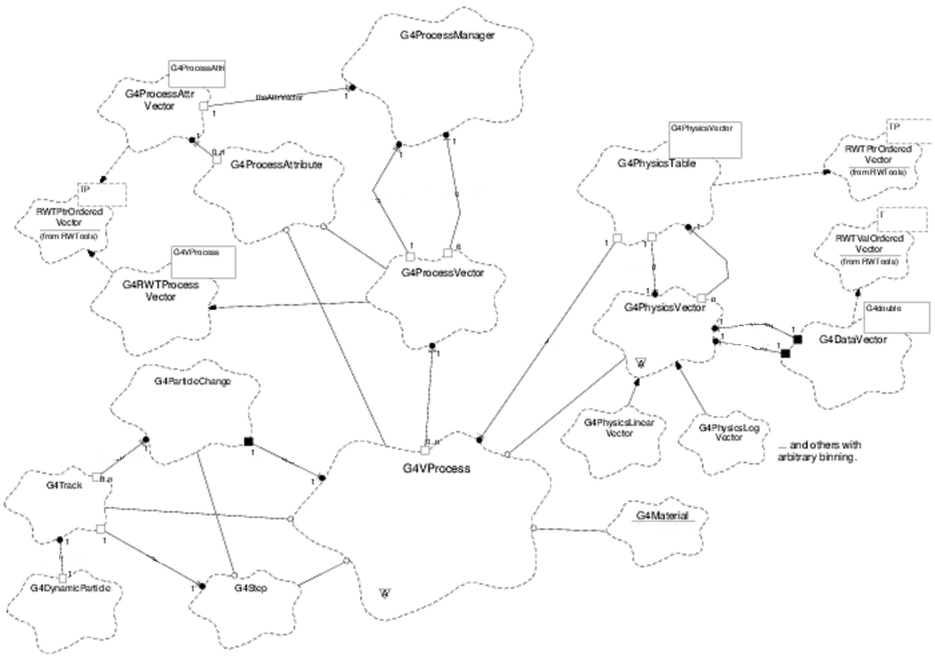


Figura 3.17: Schema del dominio Process.

3.2.7 User Interface

Geant4 permette all'utente di interagire con il programma di simulazione attraverso l'uso di comandi che possono riguardare la geometria del rivelatore, il fascio o i processi fisici attivati. Esistono comandi già implementati, divisi in directory in base alle funzionalità loro assegnate. L'utente può anche definirne, creando una sua classe *Messenger* in cui caratterizza e implementa i comandi di cui necessita.
In Geant troviamo le directory:

- **/control/**: controlla l'applicazione dei comandi definiti dall'utente. Contiene i comandi *execute*, per eseguire una macro, *verbose*, per visualizzare sullo schermo le caratteristiche del comando applicato, *saveHistory*, salva la seguenza dei comandi utilizzati in un file, è associato a *stopSavingHistory*, che termina l'operazione, *manual*, per elencare sul monitor tutti i comandi e le sub-directory definite, con la spiegazione delle funzionalità.

- **/units/**: racchiude le unità di misura definite e le trasformazioni che permettono di passare dall'una all'altra. Ha un solo comando, *list*, per elencare tutte le unità di misura che possono essere utilizzate nel programma di simulazione.

- **/tracking/**: controlla i comandi relativi agli step e alla traiettoria della particella. Contiene i comandi *abort*, per interrompere il corrente processo G4track, e *resume*, per riprenderlo, *storeTrajectory*, per memorizzare e visualizzare o no le traiettorie, e *verbose*, che indica il livello di informazioni sulle tracce delle particelle che devono essere visualizzate.

- **/event/**: gestisce i comandi relativi alla simulazione degli eventi. Contiene una sottodirectory /Stack/, responsabile della successione degli eventi. In Stack troviamo due comandi: *status*, che mostra i processi in attesa, e *clear* per eliminare la coda. Nella directory /event/ sono implementati invece i comandi *abort*, per interrompere il corrente evento, e *verbose*, che indica il livello di informazioni che l'utente vuole riguardo agli eventi.

- **/run/**: controlla i comandi legati allo svolgersi del run. Contiene una sottodirectory /particle/ dei comandi legati alle particelle definite, ai valori

di taglio dei range e ai processi fisici attivati o da attivare. I comandi principali della directory /run/ sono: *initialize*, per inizializzare il kernel di Geant4, *beamOn*, che fa partire il run, *verbose* per indicare le informazioni da visualizzare sul run, *breakAtBeginOfEvent* per fissare un punto di partenza dell'evento, e il rispettivo *breakAtEndOfEvent* per determinare il punto finale. Si aggiungono *abort* per interrompere il run, *storeRandomNumberStatus*, che memorizza lo stato del numero random in un file, e *restoreRandomNumberStatus* che ripristina lo stato di un numero random da un file.

- **/particle/**: contiene i comandi relativi alle particelle incidenti. Possiede due cartelle, /property/ e /process/ riguardanti le proprietà fisiche di ogni particella, i comandi per visualizzare le informazioni volute e attivare o disattivare determinate interazioni. In questa directory troviamo i comandi *select*, che permette di selezionare la particella, *list*, che stampa la lista delle particelle descritte in Geant, e *find*, per trovare la particella da decodificare.

- **/process/**: contiene una cartella per i processi a bassa energia, i principali comandi sono *activate* e *inactivate* per stabilire i processi da considerare e quelli da trascurare, *list* che visualizza la lista di processi implementati, e infine *verbose* per indicare il livello di informazione richiesto dall'utente.

- **/vis/**: contiene i comandi di visualizzazione, divisi per directory in base alla funzione loro assegnata. Si possono definire così lo zoom dell'immagine, l'angolazione, i colori degli oggetti descritti, etc ...

- **/gun/**: gestisce i comandi del fascio incidente, il tipo di particella (*particle*), la direzione (*direction*), il punto di partenza (*position*), l'energia cinetica (*energy*) e la polarizzazione (*polarization*). Il comando *list* stampa la lista delle particelle disponibili, *number* definisce il numero di particelle da generare e infine *random* lancia in modo casuale la particella incidente.

- **/hits/**: controlla il volume sensibile del rivelatore, attiva e disattiva parti di esso attraverso i comandi *activate* e *inactivate*, permette di elencare le componenti sensibili con *list* e infine stampa le informazioni volute dall'utente definendo un valore per il comando *verbose*.

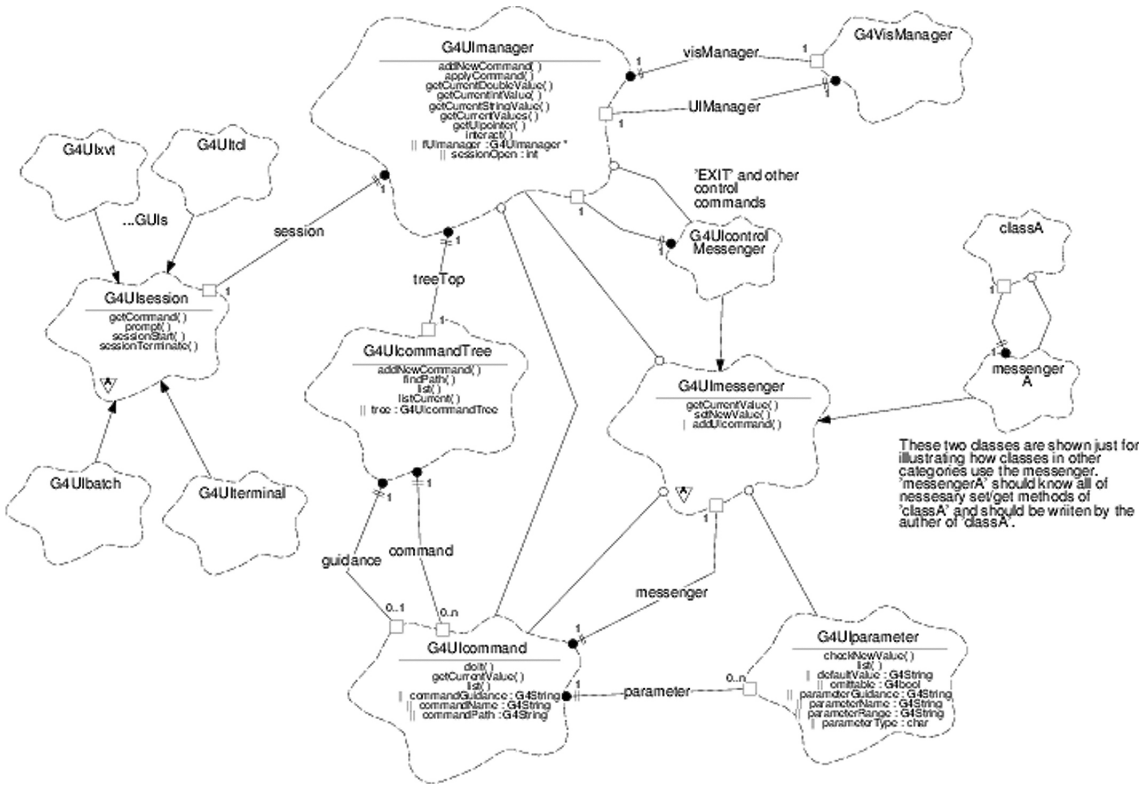


Figura 3.18: Schema del dominio G4UIMessenger.

L'utente può costruire dei comandi legati al suo programma di simulazione, attraverso le stesse classi utilizzate da Geant4. La classe *G4UIdirectory* permette di creare delle nuove directory o sottodirectory, il costruttore richiede come parametro il percorso voluto.
Un comando viene definito in base al tipo di parametro che richiede, possiamo quindi utilizzare *G4UIcmdWithABool* quando viene richiesto un valore booleano, *G4UIcmdWithAnInteger* quando si vuole un intero, *G4UIcmdWithADouble* per i double, *G4UIcmdWithAString* per le stringhe, *G4UIcmdWith3Vector* per i vettori. A queste si aggiungono due classi utilizzate nel caso che si voglia il parametro accompagnato dall'unità di misura, rispettivamente *G4UIcmdWithADoubleAndUnit* e *G4UIcmdWith3VectorAndUnit*. Per ogni comando creato, l'utente è tenuto a specificare le sue caratteristiche e le sue funzionalità.

I metodi che è possibile utilizzare sono:

⋆ *SetGuidance(char*)* visualizza una breve descrizione del comando a cui è associato.

⋆ *SetRange(char*)* definisce un range per i possibili valori associati al comando, lancia un messaggio di avviso nel caso questi non appartengano all'intervallo stabilito.

⋆ *SetParameterName(char*, G4bool)* permette di fissare il nome del comando e di stabilire se è necessario un parametro, "true" se questo può essere omesso, "false" se è richiesto.

⋆ *SetDefaultValue(G4bool/G4double/G4String/...)* per definire il valore di default.

⋆ *SetUnitCategory(char*)* definisce la categoria di appartenenza del parametro (lunghezza, peso,...)

⋆ *SetNewValue(G4UIcommand*, G4String)* viene implementato per associare ad ogni comando la sua funzione, un metodo della sopraclasse.

⋆ *GetNewDoubleValue()* viene chiamato all'interno del metodo precedente, serve ad assegnare il parametro del comando come nuovo valore.

Riportiamo un esempio di implementazione di un comando, utilizzato in questo lavoro di tesi per la simulazione della camera Farmer e della camera Monitor a Pixel, che permette di stabilire la grandezza della sezione trasversale del fascio:

```
// costruttore del comando
PrimaryGeneratorMessenger::PrimaryGeneratorMessenger(
        PrimaryGeneratorAction* prGenAct):primaryGen(prGenAct)
{
    G4UIdirectory* fascioDir = new G4UIdirectory(``/fascio/'');
    fascioDir->SetGuidance(``Particle control commands.'');

    G4UIcmdWithADoubleAndUnit* CampoCmd =
        new G4UIcmdWithADoubleAndUnit(``/fascio/campo'',this);
    CampoCmd->SetGuidance(``Set field'');
    CampoCmd->SetParameterName(``campo'',false);
    CampoCmd->SetRange(``campo>=0'');
    CampoCmd->SetUnitCategory(``Length'');
    CampoCmd->AvailableForStates(Idle);
}

// distruttore del comando
PrimaryGeneratorMessenger::~PrimaryGeneratorMessenger()
{
    delete CampoCmd;
}

// implementazione di SetNewValue()
void PrimaryGeneratorMessenger::SetNewValue(G4UIcommand* command,
G4String newValue)
{
    if(command == CampoCmd){
        primaryGen->SetCampo(CampoCmd->GetNewDoubleValue(newValue));
    }
}
```

3.2.8 Visualization

Il dominio relativo alla visualizzazione grafica mette a disposizione pacchetti software utilizzati per facilitare l'utente nel programma di simulazione e nel seguire lo svolgersi del run. Possono essere visualizzati i componenti geometrici del rivelatore, l'intera struttura contenuta nel volume madre o separatamente i volumi solido, logico e fisico di ciascun componente, le traiettorie delle particelle incidenti e di quelle prodotte. L'utente può aggiungere strutture di supporto come linee di guida e assi cartesiani, per meglio orientarsi nel disegno in 3D, e testi e commenti per la descrizione della simulazione.
La classe *G4VisAttributes* mette a disposizione dei campi per differenziare e caratterizzare le forme da visualizzare, non contiene informazioni geometriche, sulla posizione o l'orientamento, ma solo di grafica, come il colore, la visibilità, etc ...
L'utente può definire un oggetto visibile o invisibile, attraverso il metodo *SetVisibility(G4bool)* che chiede rispettivamente un parametro "true" nel primo caso (valore di default), "false" nel secondo.
Per colorare un oggetto si può utilizzare la classe *G4Colour*, questa dispone di quattro sorgenti di colore (RGBA = red, green, blue e α), che vengono mescolati, assegnando loro dei valori da 0 a 1, per ottenere tutte le tonalità. α rappresenta l'opacità ed è posta come valore di default a 1.

G4Colour	white	()
G4Colour	white	(1.0, 1.0, 1.0)
G4Colour	gray	(0.5, 0.5, 0.5)
G4Colour	black	(0.0, 0.0, 0.0)
G4Colour	red	(1.0, 0.0, 0.0)
G4Colour	green	(0.0, 1.0, 0.0)
G4Colour	blue	(0.0, 0.0, 1.0)
G4Colour	cyan	(0.0, 1.0, 1.0)
G4Colour	magenta	(1.0, 0.0, 1.0)
G4Colour	yellow	(1.0, 1.0, 0.0)

Figura 3.19: Tabella dei colori possibili e dei valori numerici per ottenerli.

3.3 File macro

Implementata l'intera simulazione, il programma viene compilato attraverso il comando `gmake`, che crea i collegamenti a Geant e i file .d e .o, necessari all'eseguibile. Successivamente il programma può essere ricompilato con `gmake NomeEseguibile`, utilizzando lo stesso nome contenuto nel GNUmakefile. Questo permette di avere l'eseguibile anche nella directory corrente. Per lanciare una simulazione serve un macro file, contenente i comandi di visualizzazione e del fascio, quello di default da creare deve essere chiamato *prerun.mac*.
Viene riportato come esempio il prerun della simulazione della camera monitor a pixel, contenente comandi standard di Geant e comandi solo della camera.

```
####   prerun della simulazione della Camera Monitor a Pixel      ####

#########   COMANDI DI GEANT4   ##########
#  livelli di informazione
 /control/verbose 1
 /run/verbose 1
 /event/verbose 1
 /tracking/verbose 1

#  visualizzazione camera
 /vis/scene/create
 /vis/sceneHandler/create OGLIX
 /vis/viewer/create
 /vis/camera/reset
 /vis/camera/viewpoint 30. 30.
 /vis/clear/view
 /vis/draw/current
 /vis/draw/axes 0. 0. 0. 10000
 /vis/show/view

# visualizzazione traiettorie
 /tracking/storeTrajectory 1
```

```
#########   COMANDI DELLA CAMERA   #######
#  Materiali
 /camera/setDeptMat  Plexiglass
 /camera/setBackMat  Plexiglass
 /camera/setGapMat   Vetronite
 /camera/setCylMat   Air

#  Dimensioni
 /camera/setDept    9.0  mm
 /camera/setBack    10.0   cm
 /camera/setRaggio  3.0   mm
 /camera/setGap     3.0   mm
 /camera/update

#########   COMANDI DI GEANT4   ##########
#  ricostruzione della scena
 /vis~/clear/scene
 /vis~/clear/view
 /vis~/draw/current

#  fascio
 /gun/particle gamma
 /gun/energy 6. MeV
 /gun/theta 0.0 deg
 /gun/campo 5.0 cm

#########   COMANDI DELLA CAMERA   #######
#  nome dell'istogramma
 /histo/sethistname prerun.hbook

#########   COMANDI DI GEANT4   ##########
#  inizia il run
 /run/beamOn 1000

##########################################
```

Capitolo 4

La camera monitor a pixel

In questo capitolo viene descritta la camera monitor a pixel, studiata per la misura della dose e la visualizzazione in tempo reale di campi di radiazione prodotti da fasci terapeutici, progettata e realizzata nell'ambito del progetto TERA. Vengono presentate e commentate alcune parti fondamentali del programma di simulazione. Questo e' stato implementato in quanto le misure ottenute dalla camera a pixel non erano in accordo con quelle di altri dosimetri standard. Le modifiche apportate direttamente sulla camera nel corso dello studio e introdotte poi nel programma di simulazione hanno determinato enormi miglioramenti. L'accordo ottenuto soprattutto nei profili di dose è mostrato da alcuni confronti fra i dati della simulazione e quelli sperimentali misurati con la camera a pixel e con la camera Farmer.

Introduzione

La misura dell'energia depositata da un fascio di particelle per ottenere dei valori di dose assorbita costituisce uno dei compiti principali della fisica medica. Vengono quindi utilizzati diversi tipi di rivelatori, tra i quali i contatori, che raccolgono il segnale, provocato dalla radiazione attraverso un circuito elettronico. Questi si differenziano per *sensibilità*, la capacità di produrre un segnale per un tipo di radiazione, *funzione di risposta*, lo spettro di impulsi ottenuto quando il rivelatore è colpito da n particelle di energia E, *risoluzione*, cioè quanto il rivelatore è preciso nell'informazione richiesta (risoluzione temporale o spaziale), *tempo di risposta*, il tempo che intercorre fra il passaggio della particella e l'uscita del segnale, e infine per l'*efficienza*, il rapporto fra le particelle rivelate e quelle incidenti.

Fra i rivelatori utilizzati, analizziamo le camere a ionizzazione e il loro funzionamento, in questo settore infatti si inserisce la camera monitor a pixel, studiata in questa tesi.

4.1 Camera a ionizzazione

Le camere a ionizzazione vengono utilizzate per la rivelazione di molte particelle, e quindi per la misura di proprietà globali del campo di radiazione, come il flusso di energia e la dose assorbita. Esse misurano la ionizzazione totale prodotta in un gas dal passaggio di particelle, attraverso due elettrodi conduttori che raccolgono il segnale.

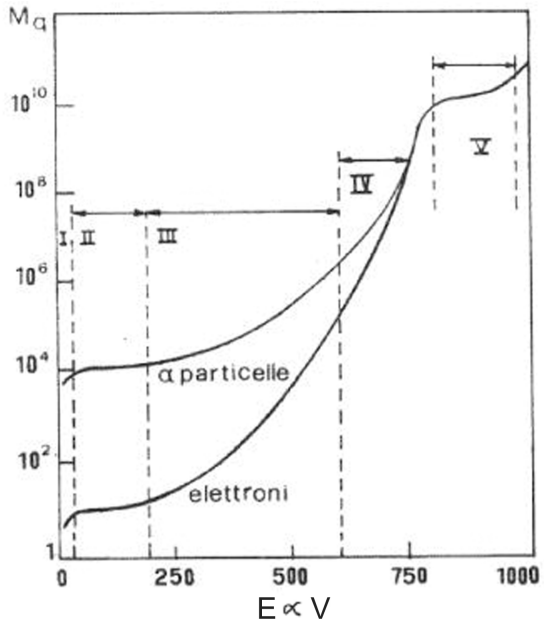


Figura 4.1: Andamento del segnale in uscita in funzione del potenziale applicato agli elettrodi.

L'impulso elettrico prodotto dipende dal potenziale applicato fra anodo e catodo, come si vede dalla fig. 4.1.

Per $V_i = 0$ non si ha il campo e gli ioni prodotti si ricombinano fra loro, così nessuna carica arriva sugli elettrodi e non si ha un segnale in uscita. Questa ricombinazione viene detta iniziale, quando la coppia ione-elettrone prodotta dalla radiazione si ricombina in un atomo neutro. Questa è tanto più probabile quanto più è piccola l'intensità del campo che agisce sugli ioni e quanto più è lungo il tempo in cui ione ed elettrone restano entro una distanza r. Esiste anche una ricombinazione generale, quando ioni prodotti in processi fisici diversi si ricombinano, migrando verso gli elettrodi, sotto l'effetto di un campo. Questo tipo di ricombinazione risulta invece proporzionale al numero di ioni presenti e quindi all'intensità del campo di radiazione[14].
Possiamo dividere la curva in cinque settori:

- **zona di ricombinazione**: in questo settore, aumentando V_i, aumenta anche il segnale uscente, il processo fisico più rilevante è la ricombinazione con M, fattore di moltiplicazione, che è inferiore a 1;

- **zona di ionizzazione**: questa è caratterizzata da un plateau, in cui la raccolta delle cariche è indipendente dal campo applicato e l'efficienza è elevata, il fenomeno fisico principale è la ionizzazione e il fattore di moltiplicazione per gli elettroni vale circa 1;

- **zona di proporzionalità**: in essa, aumentando il campo elettrico, gli elettroni generati dalla prima ionizzazione acquistano l'energia necessaria per ionizzare a loro volta, producendo una moltiplicazione di cariche che va ad aumentare il segnale in uscita in modo proporzionale all'energia persa;

- **zona a geiger limitato**: in questa regione il campo è talmente alto che il segnale cresce in modo non più proporzionale all'energia persa. Diventano importanti gli urti fra gli ioni, che dipendono dalla pressione considerata;

- **zona geiger**: il campo è così alto che si generano delle valanghe di elettroni, poichè quelli prodotti dal passaggio della radiazione ionizzano a loro volta, generando elettroni tanto energetici da far si che il processo si ripeta più volte. Si crea così una specie di goccia, alla base della quale si accumulano gli ioni positivi, che viaggiano con velocità minore, mentre sulla punta prevalgono gli elettroni. All'interno della goccia si crea un campo E_{int}, quando questo eguaglia il campo applicato, la valanga si arresta, altrimenti se ne

innescano delle altre fino ad ottenere una scarica sull'anodo. Il fattore di moltiplicazione M vale in questa zona circa 10^{10}.

La camera a ionizzazione viene utilizzata nella zona di ionizzazione, perchè si vuole raccogliere solo il segnale dovuto alla ionizzazione primaria e non il contributo della moltiplicazione[1]. Esistono camere cilindriche, sferiche o a facce piane parallele, in base alla forma geometrica degli elettrodi. La loro velocità di raccolta è determinata dalla mobilità degli ioni, definita come il rapporto fra la velocità di questi e il campo presente nella camera. Nell'aria in condizioni standard la mobilità ionica vale 1 cm/s per $Volt/cm$. Un parametro importante delle camere a ionizzazione è il volume di raccolta della carica, determinato dagli elettrodi di guardia, che delimitano le linee del campo elettrico.

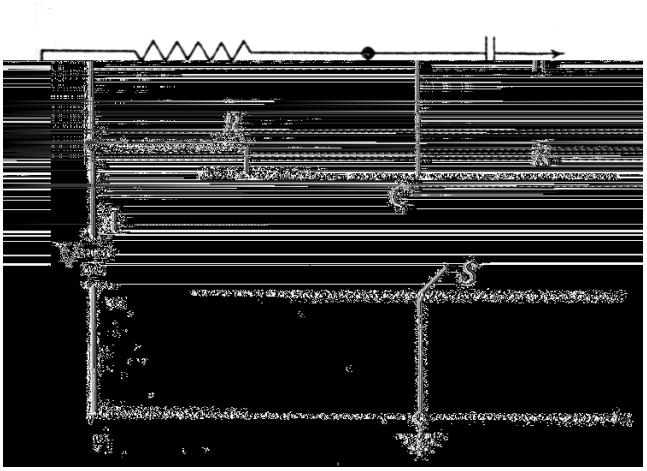


Figura 4.2: Schema di una camera a ionizzazione a elettrodi paralleli. R indica l'anello di guardia, S la sorgente di radiazioni, V il potenziale applicato per generare il campo, C l'elettrodo di raccolta, da cui si estrae il segnale elettrico.

[1]Per formare una coppia di ioni in aria occorrono circa 35 eV, dunque, se una radiazione primaria perde 1 MeV nella camera, verranno prodotte $2.86 \cdot 10^4$ coppie di ioni, corrispondenti ad una carica elettrica di $4.6 \cdot 10^{-15}$ C. (Vedi bibliografia[1])

Misura della carica raccolta

Posta la capacità della camera pari a C e la differenza di potenziale come V_0, si può calcolare la carica Q_0 depositata sugli elettrodi:

$$Q_0 = CV_0$$

Quando delle particelle attraversano il volume sensibile della camera generano degli ioni che, sotto l'effetto del campo, migrano verso gli elettrodi, diminuendo la differenza di potenziale V_0. Dalla misura del nuovo potenziale V si calcola la carica ancora depositata sugli elettrodi e quindi gli ioni raccolti:

$$Q_0 - Q = C(V_0 - V)$$

Per poter utilizzare questo metodo, il potenziale V deve ancora appartenere al plateau in cui lavora la camera. Per aumentare l'intervallo di funzionamento della stessa si deve aumentare la capacità della camera, ad esempio connettendo in parallelo una capacità nota.
Le camere che utilizzano questo metodo sono dette *a condensatore*.

4.2 Camera Monitor a Pixel

Uno degli scopi di questa camera, studiata per l'utilizzo in radioterapia, è quello di fornire in tempo reale la dose somministrata al paziente durante l'irraggiamento, la distribuzione trasversale del fascio e i parametri caratterizzanti il campo di radiazione. Può essere inoltre usata per la verifica dei piani di trattamento.

Si tratta di una camera a ionizzazione a facce piane e parallele, con anodo segmentato. Il catodo è costituito da un foglio di mylar alluminizzato di spessore pari a 100 μm e da una pellicola di rame di 35 μm. L'anodo è formato dallo stesso tipo di foglio del catodo sulle cui facce però sono depositati due strati di rame. La faccia rivolta verso il catodo viene segmentata, utilizzando tecniche litografiche, in modo da ottenere una matrice di $(32x32)$ pixels. Ogni pixel ha le dimensioni di $(7.34\ x\ 7.34)\ mm^2$ ed è distanziato dal successivo da 0.16 mm. Lo spessore sensibile, fra il catodo e l'anodo, è costituito da una griglia di vetronite forata.
Si era infatti riscontrato che il profilo di dose ottenuto dalla camera con un gap

di aria era superiore ai bordi del campo rispetto a misure effettuate con camere standard, ad esempio con la Farmer. L'introduzione di questa lastra, i cui fori definiscono il volume sensibile, ha migliorato di molto l'accordo dei profili (vedi ultimo paragrafo di questo capitolo). Ad ogni foro, il cui diametro è variato nel corso dello studio, corrisponde un pixel. Ogni componente della camera è separata dalle altre da uno strato di vetronite.
Per poter effettutare dei confronti con il fantoccio, vengono posizionati sopra e sotto la camera degli spessori variabili di plexiglass, rispettivamente chiamati depth e back.
Nell'apparato sperimentale ogni pixel dell'anodo è collegato con l'elettronica di front-end e raccoglie le cariche prodotte dalla radiazione nel volume sensibile che gli compete.

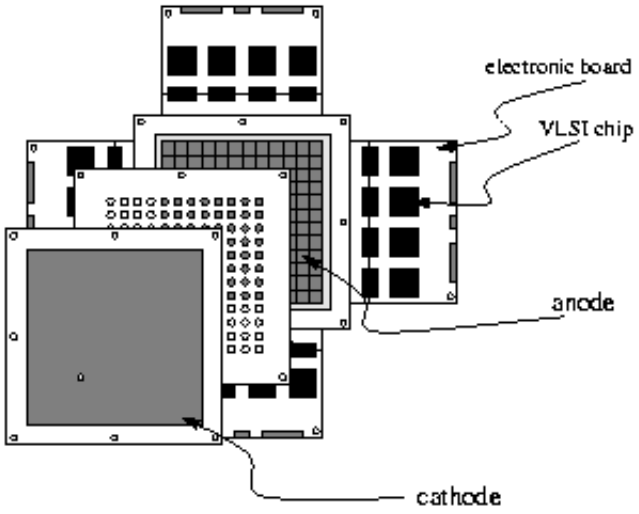


Figura 4.3: Rappresentazione schematica della camera monitor a pixel. Sono rappresentati il catodo, l'anodo segmantato e, fra questi, la griglia forata. Le schede ai quattro lati servono per la ricezione del segnale.

Figura 4.4: Vista frontale della Camera Monitor a Pixel. Mancano le schede di acquisizione dati.

Figura 4.5: Foto del retro della Camera Monitor a Pixel. Si nota la struttura necessaria a sorreggere le schede di acquisizione, l'anodo, il catodo e la griglia di vetronite.

4.3 Lo scopo della simulazione

La simulazione tende a riprodurre il più fedelmente possibile le condizioni reali dell'esperimento per poter anticipare qualitativamente i risultati che si potranno ottenere. A questo scopo è stata implementata la simulazione della camera monitor a pixel, in modo da poter 'prevedere' miglioramenti o peggioramenti dovuti ad una modifica di qualche elemento della camera stessa. Il confronto fra i dati ottenuti dalla simulazione e quelli sperimentali in una situazione semplice e in condizioni standard è la verifica che la simulazione sia affidabile e riproduca in modo attendibile l'esperimento reale. Quando questo è stato accertato, il programma viene eseguito in differenti modalità, per cercare i valori ottimali dei parametri caratterizzanti la camera. Se è presente una buona collaborazione fra simulazione e presa dati, si ottimizza il tempo dell'esperimento, evitando misurazioni inutili, in quanto i dati del MonteCarlo indicano gli sviluppi da seguire. In realtà però la simulazione è generalmente lunga se si vogliono ottenere risultati affetti da un errore statistico circa dell'1%. Vanno quindi selezionati con attenzione e discriminazione i runs da eseguire, evitando le situazioni che danno più rapidamente risultati attraverso la misura sperimentale.

4.4 Il programma di simulazione

La simulazione della camera in esame è centrata su una funzione *main()*, che gestisce le chiamate di tutte le altre classi, implementata nel modulo principale *Cmp.cc*. Essa utilizza un'istanza della classe *G4RunManager* per gestire le procedure di inizializzazione, l'avvio e il termine dei run e per richiamare gli oggetti delle principali classi utilizzate. Ognuna di queste è definita, con i suoi metodi e le sue variabili, in un file di estensione '.hh', contenuto in una directory *include*, e implementata in un file omonimo di estensione '.cc', contenuto in una directory *src*.
Le classi che sono state definite sono:

- **CameraDetectorConstruction**, in cui vengono definiti gli elementi geometrici e i materiali del rivelatore, viene dichiarato il volume sensibile e implementati i metodi necessari per variare i parametri della camera;

- **CameraDetectorMessenger** permette di definire dei comandi interattivi, che possono servire all'utente per cambiare, in modo semplice e rapido, alcuni parametri geometrici della camera;

- **CameraPrimaryGeneratorAction** è la classe che definisce il fascio, il tipo e il numero di particelle da generare e il profilo che questo deve assumere;

- **CameraPrimaryGeneratorMessenger** permette di gestire, attraverso dei comandi, i parametri riguardanti il fascio e le particelle che lo compongono;

- **CameraPhysicsList** richiama tutte le classi di Geant in cui sono implementati i processi fisici che servono alla simulazione, definisce le particelle che devono essere prese in considerazione e per ognuna di esse i cut in range da utilizzare;

- **CameraSteppingAction** divide la traiettoria della particella in step e calcola l'energia depositata per ognuno di essi nel volume sensibile;

- **CameraEventAction** è la classe che gestisce lo svolgersi dell'evento, calcola l'energia accumulata in esso e definisce le caratteristiche per la sua visualizzazione sullo schermo;

- **CameraRunAction** gestisce invece lo sviluppo del run, somma l'energia depositata per ogni evento e permette di costruire un istogramma 3D e una n-tupla dove vengono memorizzati i dati della simulazione;

- **CameraRunMessenger** definisce un comando per stabilire il nome dell'istogramma, questo permette di poter eseguire dei run successivi senza sovrascrivere i dati;

- **CameraTrackerHit** e **CameraTrackerSD** inizializzano e gestiscono la parte sensibile del rivelatore;

- **CameraVisManager** richiama tutti i pacchetti grafici collegati a Geant per la visualizzazione del rivelatore e delle particelle durante la simulazione.

Durante questo lavoro di tesi il programma è stato modificato più volte per cercare di riprodurre al meglio le condizioni sperimentali e introdurre le modifiche che venivano apportate alla camera. In ogni simulazione presentata e in ogni confronto con i dati sperimentali verranno quindi definiti i valori dei parametri caratterizzanti gli elementi della camera.

4.4.1 Main()

Nella funzione *main()* viene definito il metodo per la generazione di numeri random, attraverso la classe di CLHEP *RandFlat.h*, e il seed con il quale l'algoritmo inizia. Si creano le istanze delle classi G4RunManager, per la gestione del run, G4UImanager, per l'interfaccia grafica, CameraDetectorConstruction, CameraRunAction e CameraEventAction, necessarie per la simulazione, e si applicano i metodi di inizializzazione. Al main può essere passato come parametro un file macro, che contiene le righe di comando da eseguire (di default viene eseguito il file 'prerun.mac', contenuto nella directory corrente).

Riporto il codice commentato:

// si includono le classi che vengono utilizzate nella funzione main()

```
#include ‘‘G4RunManager.hh’’
#include ‘‘G4UImanager.hh’’
#include ‘‘G4ios.hh’’
#include ‘‘G4UIterminal.hh’’
#include ‘‘CLHEP/Random/RandFlat.h’’
#include ‘‘CameraDetectorConstruction.hh’’
#include ‘‘CameraPhysicsList.hh’’
#include ‘‘CameraPrimaryGeneratorAction.hh’’
#include ‘‘CameraSteppingAction.hh’’
#include ‘‘CameraEventAction.hh’’
#include ‘‘CameraRunAction.hh’’
#ifdef G4VIS_USE
#include ‘‘CameraVisManager.hh’’
#endif

int main(int argc, char** argv)
```

```
{
// scelta del numero random
HepRandom::setTheEngine(new RanecuEngine);
long seeds[2];
int index = 0;
HepRandom::getTheTableSeeds(seeds, index);
HepRandom::setTheSeeds(seeds, index);
HepRandom::showEngineStatus();

// si creano le istanze
G4RunManager* runManager = new G4RunManager;
CameraDetectorConstruction* detector = new CameraDetectorConstruction;
CameraRunAction* run = new CameraRunAction;
CameraEventAction* event = new CameraEventAction(run);
G4UImanager* UI = G4UImanager::GetUIpointer();

// si inizializzano il rivelatore, i processi fisici e la visualizzazione
runManager->SetUserInitialization(detector);
runManager->SetUserInitialization(new CameraPhysicsList);
#ifdef G4VIS_USE
G4VisManager* visManager = new CameraVisManager;
visManager->initialize();
#endif

// impostazione azione utente
runManager->SetUserAction(new CameraPrimaryGeneratorAction(detector));
runManager->SetUserAction(event);
runManager->SetUserAction(run);
runManager->SetUserAction(new CameraSteppingAction(detector, event));

// inizializzazione kernel Geant4
runManager->initialize();

// gestione parametri
if(argc == 1)
```

```
{
G4UISession* session = new G4UITerminal;
UI->ApplyCommand(‘‘/control/execute prerun.mac’’);
session->SessionStart();
delete session;
}
else
{
G4UISession* session1 = new G4UITerminal;
G4String command = ‘‘/control/execute’’;
G4String fileName = argv[1];
UI->ApplyCommand(command + fileName);
session1->SessionStart();
delete session1;
}

// rimozione istanze create
#ifdef G4VIS_USE
delete visManager;
#endif
delete runManager;
return 0;
}
```

4.4.2 Geometria

Per simulare in Geant4 la geometria della camera è stato necessario definire inizialmente un *WorldVolume*, di dimensioni (2.5 x 2.5 x 2.5) m^3, in cui fossero contenuti tutti gli elementi geometrici del rivelatore e l'origine del fascio. La camera è stata divisa in moduli elementari, caratterizzati da una forma e un materiale, posizionati nello spazio in relazione al centro degli assi. Sono stati implementati i pixel di rame, gli anodi e i catodi, due di rame e due di vetronite, e i due spessori di plexiglass posti sopra e sotto la camera.

Per implementare i materiali che servivano alla simulazione, si sono dovuti definire gli elementi di cui erano composti: Azoto, Idrogeno, Ossigeno, Silicio e Carbonio. Riporto nella tabella 4.6 le loro proprietà.
Gli elementi geometrici della camera sono presentati nella tabella (4.7), caratterizzati dalle dimensioni e dai materiali di cui sono costituiti.

Elementi	Simbolo	Peso Molecolare (a)	num p+ (Z)
azoto	N	14,01 g/mole	7
idrogeno	H	1,01 g/mole	1
ossigeno	O	16,00 g/mole	8
silicio	Si	28,09 g/mole	14
carbone	C	12,01 g/mole	6

Figura 4.6: Tabella delle proprietà fisiche degli elementi descritti.

Volumi	Dimensioni	Materiali
World	(2,5 x 2,5 x 2,5) m3	aria
pixel	(7,34 x 7,34 x 0,035) mm3	rame
anodoV	(170 x 170 x 0,1) mm3	mylar
anodoC	(170 x 170 x 0,035) mm3	rame
catodoV	(170 x 170 x 0,1) mm3	mylar
catodoC	(170 x 170 x 0,035) mm3	rame
gapPixel	(7,5 x 7,5 x 3) mm3	vetronite
cylinder	(6 x 6 x 3) mm3	aria
dept	(180 x 180 x 9) mm3	plexiglass
back	(180 x 180 x 100) mm3	plexiglass

Figura 4.7: Tabella dei volumi descritti e delle loro dimensioni.

I materiali descritti sono aria, rame, plexiglass e vetronite, ottenuta dalla miscela di quarzo ed epoxy. Il codice che li implementa è il seguente:

```
void CameraDetectorConstruction::DefineMaterials()
{
...
// definizione variabili
G4double a, density, z;
G4String name;

// rame
density =8.960*g/cm3;
a = 63.55*g/mole;
G4Material* Cu = new G4Material(name="Copper",z= 29., a, density);

// aria
density = 1.290*mg/cm3;
G4Material* Air = new G4Material(name="Air", density, ncomponents=2);
Air->AddElement(N, fractionmass=0.7);
Air->AddElement(O, fractionmass=0.3);

// plexiglass
density = 1.19*g/cm3;
G4Material* Plexiglass = new G4Material(name="Plexiglass", density,
ncomponents=3);
Plexiglass->AddElement(C, fractionmass=0.60);
Plexiglass->AddElement(O, fractionmass=0.32);
Plexiglass->AddElement(H, fractionmass=0.08);

// quarzo
density = 2.200*g/cm3;
G4Material* SiO2 = new G4Material(name="quartz", density, ncomponents=2);
SiO2->AddElement(Si, natoms=1);
SiO2->AddElement(O , natoms=2);
```

```
// epoxy
density = 1.00 *g/cm3;
G4Material*epoxy= new G4Material(name="epoxy", density, ncomponents=3);
epoxy->AddElement(C, natoms=1);
epoxy->AddElement(H, natoms=1);
epoxy->AddElement(O, natoms=1);

// vetronite
density = 1.7*g/cm3;
G4Material* G10 = new G4Material(name="G10", density, ncomponents=2);
G10->AddMaterial(SiO2, fractionmass=60.*perCent);
G10->AddMaterial(epoxy,fractionmass=40.*perCent);

// stampa la tavola dei materiali
G4cout<< *(G4Material::GetMaterialTable())<<endl;
...
}
```

La griglia, i cui fori costituiscono la parte sensibile del rivelatore, è stata implementata costruendo dei cubi, di dimensioni $(7.5\ x\ 7.5\ x\ 3)\ mm^3$, a cui è stato sottratto un cilindro, di diametro $d\ =\ 6\ mm$ e altezza $h\ =\ 3\ mm$. Il solido così ottenuto, chiamato GapPixel, è stato definito di vetronite. Nel foro è stato inoltre inserito un cilindro di aria, di dimensioni identiche al precedente. Per posizionare i GapPixel l'uno vicino all'altro, lungo i due assi cartesiani, si sono dovuti utilizzare due cicli *for* annidati, definendo le variabili di posizione, 'GapPixelPosX' e 'GapPixelPosY', come funzioni del numero di pixel e della distanza fra questi.
Nel costruttore della classe sono stati definiti i valori, presi dal programma come valori di default, delle variabili che caratterizzavano le dimensioni degli elementi geometrici del rivelatore. Questi ultimi sono stati implementati nel metodo *ConstructCamera()*.

```
CameraDetectorConstruction::CameraDetectorConstruction()
{
...
```

```
// pixel
PixelDist = 7.5*mm;
NbOfPixel = 32;

// cilindro di aria
CylinderInnerRadius = 0.0*mm;
CylinderOuterRadius = 3.0*mm;
CylinderStartAngle = 0.0*deg;
CylinderSpanningAngle = 360.0*deg;
CylinderMaterial = Air;

// cubo forato di vetronite
GapPixelThickness = 3.0*mm;
GapPixelPosZ = 0.0*mm;
GapPixelMaterial = G10;
...
}

G4VPhysicalVolume* CameraDetectorConstruction::ConstructCamera()
{
//--- Descrizione della griglia

for(G4int l=0;l<NbOfPixel;l++) { // ciclo sulle x
for(G4int m=0;m<NbOfPixel;m++) { // ciclo sulle y
G4int n=l*NbOfPixel+m; // definiscono un numero
G4int k=l*NbOfPixel+m; // indicativo di ogni pixel

// descrizione di un cubo e un cilindro solidi
// fra cui viene effettuata una sottrazione booleana
// per ottenere il GapPixel di Vetronite

// --- AriaPixel
solidAriaPixel[n] = new G4Box("boxAriaPixel",
AriaPixelSizeXY/2,AriaPixelSizeXY/2,AriaPixelThickness/2);
```

```
// --- CyliPixel
solidCyliPixel[n] = new G4Tubs("tubsCyliPixel",
CyliPixelInnerRadius,CyliPixelOuterRadius,
CyliPixelThickness/2,
CyliPixelStartAngle,CyliPixelSpanningAngle);

// --- GapPixel
G4double GapPixelPosY = ( -((NbOfPixel/2-0.5)*(PixelDist)) +
                            + m*PixelDist);
G4double GapPixelPosX = ( -((NbOfPixel/2-0.5)*(PixelDist)) +
                            + l*PixelDist);

solidGapPixel[n] = new G4SubtractionSolid("Box-Tubs",
solidAriaPixel[n],
solidCyliPixel[n]);

// solo del GapPixel si definisce volume logico e fisico
logicGapPixel[n] = new G4LogicalVolume(solidGapPixel[n],
GapPixelMaterial,
"logGapPixel");

phyGapPixel[n] = new G4PVPlacement(0,
G4ThreeVector(GapPixelPosX,GapPixelPosY,GapPixelPosZ),
logicGapPixel[n],
"GapPixel",
logicWorld,
false, //non compie più operazioni booleane,
n); //deriva da una op.  booleana!

// cilindro di aria che viene inserito nel GapPixel
// --- Cylinder
G4double CylinderPosY = ( -((NbOfPixel/2-0.5)*(PixelDist)) +
                            + m*PixelDist);
G4double CylinderPosX = ( -((NbOfPixel/2-0.5)*(PixelDist)) +
```

```
                                + l*PixelDist);

solidCylinder[k] = new G4Tubs("tubsCylinder",
CylinderInnerRadius,
CylinderOuterRadius,
CylinderThickness/2,
CylinderStartAngle,
CylinderSpanningAngle);

logicCylinder[k] = new G4LogicalVolume(solidCylinder[k],
CylinderMaterial,
"logCylinder");

phyCylinder[k] = new G4PVPlacement(0,
G4ThreeVector(CylinderPosX,CylinderPosY,CylinderPosZ),
logicCylinder[k],
"Cylinder",
logicWorld,
false,
k);

// fori colorati di azzurro
G4VisAttributes* CylAttr = new G4VisAttributes(G4Colour(0.5,3.0,1.0));
4 logicCylinder[k]->SetVisAttributes(CylAttr);

} // for(l)
} // for(m)

}
```

L'aria contenuta nei fori è il volume sensibile della camera, in cui calcoliamo l'energia depositata. Nel programma viene quindi assegnato il volume logico del 'Cylinder' come istanza della classe *CameraTrackerSD*. Quest'implementazione, eseguita sempre nel metodo *ConstructCamera()*, è un po' laboriosa poichè richiede di definire sia l'oggetto della classe della simulazione che tratta la parte

sensibile del rivelatore, sia quello della classe di Geant4 *G4SDManager*.

```
G4VPhysicalVolume* CameraDetectorConstruction::ConstructCamera()

{
...
// parte sensibile del rivelatore

// crea SDMan come oggetto di G4SDManager
// è un puntatore al volume sensibile
G4SDManager* SDMan = G4SDManager::GetSDMpointer();

// crea l'oggetto 'volume sensibile'
// caratterizzato dal nome 'CameraSD'
// e lo aggiunge a SDMan
G4String CameraSDname = "CameraSD";
CameraTrackerSD* aTrackerSD = new CameraTrackerSD(CameraSDname);
SDMan->AddNewDetector(aTrackerSD);

// definisce ogni foro della griglia
// come volume sensibile
G4int dimension = NbOfPixel*NbOfPixel;

for(G4int p=0;p<dimension;p++){
logicCylinder[p]->SetSensitiveDetector(aTrackerSD);
} // for(p)

...

}
```

Nella classe *CameraDetectorConstruction()* sono state implementate anche le funzioni che associano dei nuovi valori ad alcune variabili, metodi *Set*, a cui corrispondono quelle che restituiscono il valore corrente, metodi *Get*.

Ne presento qualcuna di esempio:

1) per associare un nuovo materiale alla griglia

```
void CameraDetectorConstruction::SetGapPixelMat(G4String gapMat)
{
// associa l'istanza 'material' creata al parametro del metodo
G4Material* material = G4Material::GetMaterial(gapMat);

// definisce ogni GapPixel di questo materiale
if (material)
{
GapPixelMaterial = material;

for(G4int p=0; p<dimension; p++){
logicGapPixel[p]->SetMaterial(material);
} for(p)
} if
}
```

2) per variare il raggio dei fori della griglia

```
void CameraDetectorConstruction::SetCyliPixelRadius(G4double raggio)
{
// associa il parametro alla variabile 'raggio'
CyliPixelOuterRadius = raggio;
}
```

3) per ricostruire la geometria

```
void CameraDetectorConstruction::UpdateGeometry()
{
// viene ripetuto il metodo ConstructCamera()
G4RunManager::GetRunManager()->DefineWorldVolume(ConstructCamera());
}
```

Per variare i parametri della geometria è stata definita la classe *G4DetectorConstructionMessenger*, dove sono implementati i comandi, utilizzabili dall'utente in modo interattivo, che richiamano i metodi *Set*. Può essere ad esempio variato lo spessore di plexiglass sopra la camera, con il comando "/camera/setDept", o il suo materiale, con "/camera/setDeptMat". Così anche per le dimensioni e i materiali del gap, dei fori e dello spessore back, presente sotto la camera. Tutti i comandi relativi alla geometria si trovano all'interno di una cartella "/camera/". Riportiamo l'implementazione di uno dei comandi definiti, che gestisce il raggio dei fori della griglia di vetronite:

```
// costruttore
CameraDetectorMessenger::CameraDetectorMessenger
(CameraDetectorConstruction* CameraDet)
:CameraDetector(CameraDet)
{
// crea la directory 'camera'
CameradetDir = new G4UIdirectory("/camera/");
CameradetDir->SetGuidance("Camera detector control.");

// crea e caratterizzo il comando per variare il raggio dei fori
RaggioCmd = new G4UIcmdWithADoubleAndUnit("/camera/setRaggio",this);
RaggioCmd->SetGuidance("Fissa il Raggio dei fori della groviera");
RaggioCmd->SetParameterName("Size",false);
RaggioCmd->SetRange("Size>=0.");
RaggioCmd->SetUnitCategory("Length");
RaggioCmd->AvailableForStates(Idle);
}


// distruttore
CameraDetectorMessenger::~CameraDetectorMessenger()
{
delete RaggioCmd;
}
```

```
// implementazione del comando
void CameraDetectorMessenger::SetNewValue(G4UIcommand*
command,G4String newValue)
{
if( command == RaggioCmd ){
CameraDetector->SetCyliPixelRadius(
RaggioCmd->GetNewDoubleValue(newValue));
} if
}
```

4.4.3 Fascio

La classe *CameraPrimaryGeneratorAction* genera le particelle incidenti sul rivelatore, secondo uno spettro energetico e una distribuzione spaziale, definita dal campo di radiazione. Nella simulazione della camera sono stati utilizzati fotoni da 6 MeV, per campi variabili da 25 a 225 cm^2. Dato che i fotoni non interagiscono fra di loro e la mutua interazione degli elettroni prodotti è trascurabile rispetto a quella che questi hanno con la materia, viene mandata sul rivelatore una particella per volta, invece che N fotoni contemporaneamente. In questo modo il programma riesce a elaborare i dati.
Nel costruttore della classe viene stabilito il tipo di particella voluto e i valori di default per le variabili che descrivono il numero totale di particelle, la semilarghezza del campo e l'angolazione del fascio rispetto alla superficie da irradiare. Vengono aperti i file per la memorizzazione dei dati riguardanti le particelle incidenti: 'enout.dat' relativo alla loro energia, 'x0.dat' per sapere le coordinate da cui partono e 'N.dat' per conoscere il numero di quelle che appartengono al centro del fascio o al profilo gaussiano, di cui tratteremo in seguito.

```
// definisce una variabile non legata alla classe
G4String CameraPrimaryGeneratorAction::
thePrimaryParticleName="gamma";

// costruttore
CameraPrimaryGeneratorAction::CameraPrimaryGeneratorAction
```

```
(CameraDetectorConstruction *CameraDC)
:CameraDetector(CameraDC), rndmFlag("on")
{
// stabilisce quante particelle mandare contemporaneamente
G4int n_particle = 1;
particleGun = new G4ParticleGun(n_particle);

// valori di default
L = 25.*mm;
sigma = 5.*mm;
theta = 0.0*deg;
```
$N = 10^7$;
```
// crea i file
enout.open("enout.dat");
if(!enout)cout<<"impossibile aprire enout.dat"<<endl;
out.open("x0.dat");
if(!out)cout<<"impossibile aprire x0.dat"<<endl;
Nout.open("N.dat");
if(!Nout)cout<<"impossibile aprire N.dat"<<endl;
}
```

Il metodo principale è *GeneratePrimaries()*, a cui viene passato l'evento come parametro. In esso distinguiamo tre implementazioni differenti: una riguardante lo spettro energetico dei fotoni, una sulla penombra gaussiana del fascio e l'ultima sull'angolazione che questo può avere.
Per la produzione di fotoni, nella testata di un acceleratore terapeutico, un fascio di elettroni viene indirizzato su un bersaglio, da cui si ottengono raggi γ. Questi vengono poi fatti passare attraverso un magnete per eliminare dalla traiettoria possibili elettroni residui, e in seguito in un precollimatore fisso. Per appiattire la distribuzione di intensità nel campo di radiazione viene inserito lungo il loro percorso un filtro di tungsteno, ferro e piombo.
Possiamo quindi affermare che lo spettro energetico dei fotoni sia essenzialmente dovuto a perdita di energia di bremsstrahlung da parte degli elettroni incidenti, che vengono frenati dal materiale del bersaglio. Questo è confermato dal confron-

to della curva della sezione d'urto di bremsstrahlung in funzione dell'energia del fotone uscente con i dati sperimentali provenienti da un acceleratore lineare.
Nel programma vengono caricati dei valori sperimentali da due file, 'en.dat' e 'Flusso.dat', attraverso i quali si calcola una distribuzione di probabilità $F(E)$. Successivamente si genera un numero random appartenente a questa distribuzione e, per interpolazione lineare, si ottiene il valore E_i di energia della particella da inviare sul rivelatore.

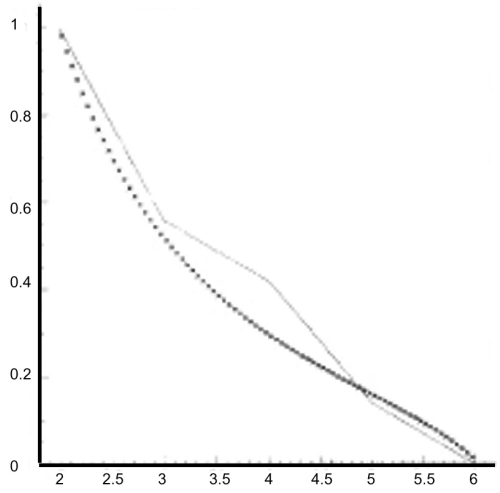


Figura 4.8: Confronto fra le code della sezione d'urto di bremsstrahlung, per elettroni incidenti di 6 MeV su Tungsteno, e dello spettro energetico di fotoni uscenti da un acceleratore lineare, con potenziale applicato Vo = 6 MV.

Il passaggio del fascio nel collimatore genera una 'penombra', cioè un numero minore di fotoni ai bordi rispetto al centro. E' stato preso in considerazione dunque un profilo del fascio piatto a cui si sono aggiunte code gaussiane ai lati, normalizzato in modo tale che il massimo della gaussiana corrispondesse all'altezza del profilo rettangolare. Il numero di particelle incidenti N è stato diviso in N_{flat}, quelle contenute in un quadrato di lato $2(L - \sigma)$, e N_{gauss}, quelle appartenenti alla gaussiana di deviazione standard σ. I valori di queste due variabili sono stati calcolati attraverso il rapporto delle aree della sezione quadrata e gaussiana del

fascio, imponendo la condizione che la loro somma fosse pari a N.

$$N = N_{flat} + N_{gauss}$$

$$\frac{N_{flat}}{N} = \frac{A_q}{A_q + A_g}$$

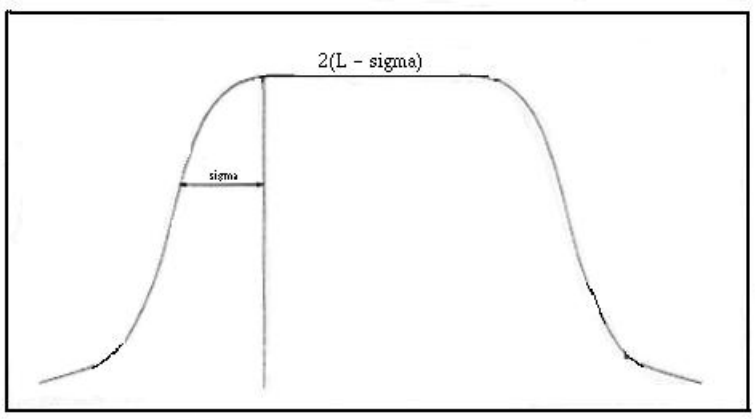


Figura 4.9: Rappresentazione del profilo del fascio utilizzato nella simulazione. La lunghezza totale del profilo è pari a $2L$. Le gaussiane sono unite a tutti quattro i lati del fascio rettangolare.

Nell'implementazione viene generato un numero random s compreso fra 0 e N, che viene confrontato con i valori calcolati N_{flat} e N_{gauss}.

- $s < N_{flat}$: la particella parte in un punto di coordinate (x_0, y_0) appartenenti alla sezione rettangolare del fascio, generate secondo una distribuzione piatta;
- $N_{flat} < s < N_{flat} + N_{gauss}/2$: la coordinata x_0 appartiene al centro del fascio, mentre y_0 ai bordi, viene quindi generato il profilo gaussiano lungo l'asse delle ordinate;
- $N_{flat} + N_{gauss}/2 < s < N$: la coordinata y_0 viene fissata secondo una distribuzione piatta, mentre x_0 secondo quella gaussiana, per generare la penombra sull'asse delle ascisse.

L'implementazione utilizza le classi di CLHEP *G4UniformRand()*, per fissare un valore al parametro s, *RandFlat* e *RandGauss* per fissare le coordinate di ogni particella generata. La posizione sull'asse z è fissata in relazione allo spessore di plexiglass presente davanti alla camera, dall'istruzione:

```
{
...
SSD = 100.*cm;
z0 = (CameraDetector->GetDeptPosZ()) -
((CameraDetector->GetDeptThickness())/2 + SSD);
...
}
```

La terza parte della classe permette di ruotare l'origine del fascio lungo una circonferenza centrata nell'origine, di raggio z0. Si considerano inizialmente le coordinate ricavate nella posizione $\theta = 0$, alle quali viene aggiunta o sottratta una quantità costante, legata al seno o coseno dell'angolo. Il discorso si ripete anche per l'impulso delle particelle incidenti, che non è più parallelo a z.

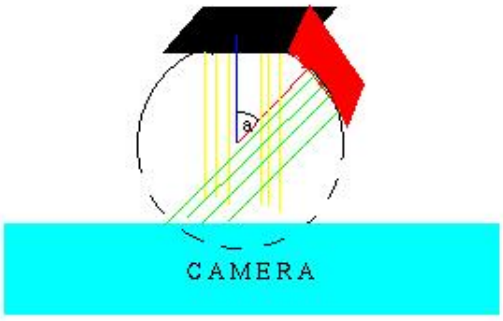


Figura 4.10: Rappresentazione di come viene ruotato il fascio e il campo di radiazione.

Vengono dichiarate le nuove coordinate e le nuove componenti dell'impulso:

```
{
...
// coordinate
G4double x1=(x0-z0*sin(theta));
G4double y1= y0;
G4double z1=(z0*cos(theta));

// impulso
G4double px1= -(sin(theta));
G4double py1= 0.;
G4double pz1= (cos(theta));
...
}
```

L'energia, la posizione iniziale e il momento vengono associati alla particella attraverso dei metodi *Set* della classe *G4ParticleGun*, di cui era stata definita un'istanza nel costruttore. Il fotone viene creato dal metodo *GeneratePrimaryVertex*, a cui viene passato l'evento come parametro, che viene richiamato più volte, per ogni particella, fino a generare il fascio.

```
{
...
particleGun->SetParticleEnergy(E_i);
particleGun->SetParticlePosition(G4ThreeVector(x1, y1, z1));
particleGun->SetParticleMomentumDirection(G4ThreeVector(px1,py1,pz1));
particleGun->GeneratePrimaryVertex(anEvent);
...
}
```

Nella classe sono stati infine implementati i metodi *Set* e *Get*, necessari per creare dei comandi interattivi, nella classe *CameraPrimaryGeneratorMessenger*, che permettessero all'utente di modificare il campo di radiazione, con "/gun/campo/", il numero di eventi da inviare sul rivelatore, con "/gun/eventi", e infine l'angolo del fascio, con il comando "/gun/theta".

4.4.4 Rappresentazione dei risultati

Al termine del run, Geant restituisce un istogramma 2D, che rappresenta la matrice di (32 x 32) pixel riempito con l'energia depositata in ciascuno di questi, espressa in MeV. L'implementazione, contenuta in *CameraRunAction*, utilizza delle classi di CLHEP, appartenenti alla directory 'Hist', con le quali vengono definiti e riempiti una n-tupla e l'istogramma.
Entrambi sono creati nel metodo *bookHisto()*, attraverso le seguenti istruzioni.

```
void CameraRunAction::bookHisto()
{
// HbookFile richiede come parametri
// una stringa (nome) e un intero (identificativo)
hbookManager = new HBookFile(histName,68);
assert (hbookManager != 0);

// nel file .hbook creato sono contenuti
// sia la ntupla, sia l'istogramma

// la ntupla è caratterizzato solo dal nome
ntuple = hbookManager->ntuple("energiadep");
assert(ntuple != 0);

// l'istogramma richiede oltre al nome,
// il numero di bins in cui dividere l'asse
// il valore minimo e il valore massimo
histo = hbookManager->histogram("edep",
32,0.0,32.0 , // sulle x
32,0.0,32.0); // sulle y
assert (histo != 0);
}
```

Questo metodo è chiamato all'inizio del run, in *BeginOfRunAction*, ma i dati vengono inseriti solo alla fine, nel metodo *EndOfRunAction*.
L'istogramma è riempito grazie ad un metodo *accumulate(x,y,En)* dalle seguenti righe di comando.

```
void CameraRunAction::EndOfRunAction(const G4Run* run)
{
...
 // ciclo su tutti i pixel
 for(G4int i=0;i<dimension;i++){

   // se l'energia depositata è ≠ 0
   if(EdepPixel[i] != 0){

      // vengono definite le coordinate
      float x=i/32+0.5;
      float y=i%32+0.5;

      if(histo){
         histo->accumulate(x,y,EdepPixel[i]);
      } // if

      // vengono salvati i dati in un file
      outEnergy<<" Pixel:  "<<i<<" En.  Dep.  "<<EdepPixel[i]<<endl;

   } // if

 } // for
...
// viene stampato l'istogramma nel file .hbook
hbookManager->write();
...
}
```

E' stato definito un comando che permette all'utente di definire il nome del file hbook creato, in questo modo possono essere effettuati dei run successivi, ciascuno dei quali crea un suo istogramma, senza che vengano sovrascritti i dati. Il comando, "/sethistname", è implementato nella classe *CameraRunMessenger* e contenuto nella directory "/histo/".

4.5 Il problema della penombra

Inizialmente la camera monitor a pixel non presentava la lastra di vetronite fra anodo e catodo, ma un gap di aria. Confrontando però i dati sperimentali ottenuti con quelli ricavati da un dosimetro standard inserito nel fantoccio ad acqua, con lo stesso campo di radiazione e utilizzando lo stesso acceleratore, si è notato che non vi era accordo soprattutto nella regione della penombra, in particolare la camera presentava una discesa meno accentuata.

La penombra è stata definita come la differenza fra le ascisse dei punti posizionati sull'asse delle ordinate rispettivamente all' 80% e al 20% del valore massimo del profilo. Per un gap di aria di 3 mm la penombra calcolata è (9.05 ± 0.28) mm. Per questo motivo si è pensato di introdurre nel gap del materiale che assorbisse parte dei fotoni laterali e si è costruita la lastra di vetronite, con fori di diametro 6 mm, che definiscono il volume sensibile della camera. I profili ottenuti con questo assetto sperimentale presentano un notevole miglioramento nelle code, come mostrano i confronti per il campo $(5x5)$ cm^2 e $(10x10)$ cm^2, a 10 cm di profondità, per fotoni ottenuti con $6MV$ di potenziale applicato.

Accertati i miglioramenti ottenuti con la lastra forata, questa configurazione della camera è stata riprodotta nel codice di simulazione, per studiare l'importanza di alcuni parametri, come il materiale di cui è costituita, il suo spessore e il diametro dei fori, sull'accordo ottenuto con i dati della camera Farmer.
Le fasi di studio di un dosimetro sono essenzialmente determinate da tre misurazioni:
1) i profili di dose,
2) lo studio della dose in funzione della profondità,
3) la misura dell'output factor.

Lo stesso studio è stato eseguito con la simulazione, sia per il confronto con i dati sperimentali, quando questi erano in accordo con quelli della Farmer, sia per l'analisi di possibili variazioni da effettuare, quando non vi era accordo.

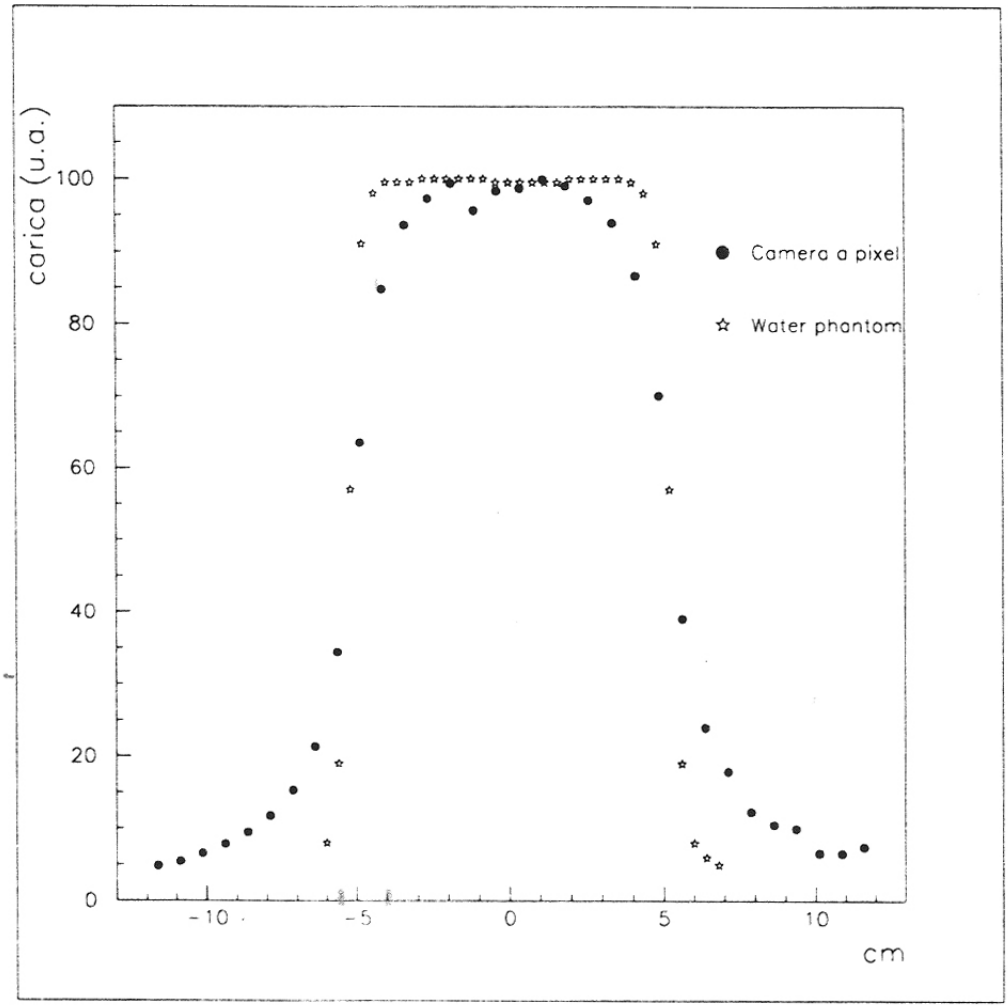
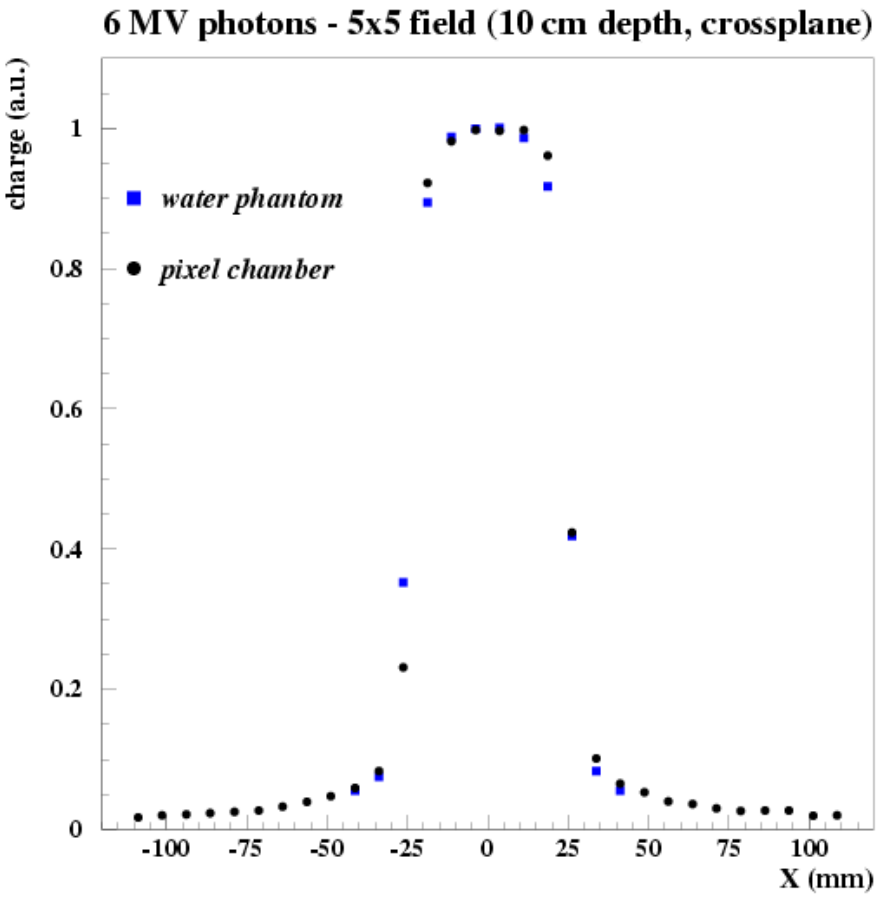


Figura 4.11: Confronto fra le code ottenute con un fantoccio ad acqua e con la camera monitor a pixel per un campo di radiazione $(10x10)\ cm^2$. Si nota che quest'ultima presenta una maggior penombra.


6 MV photons - 5x5 field (10 cm depth, crossplane)
charge (a.u.)
1
0.8
0.6
0.4
0.2
0
water phantom
pixel chamber
-100
-75
-50
-25
0
25
50
75
100
X (mm)

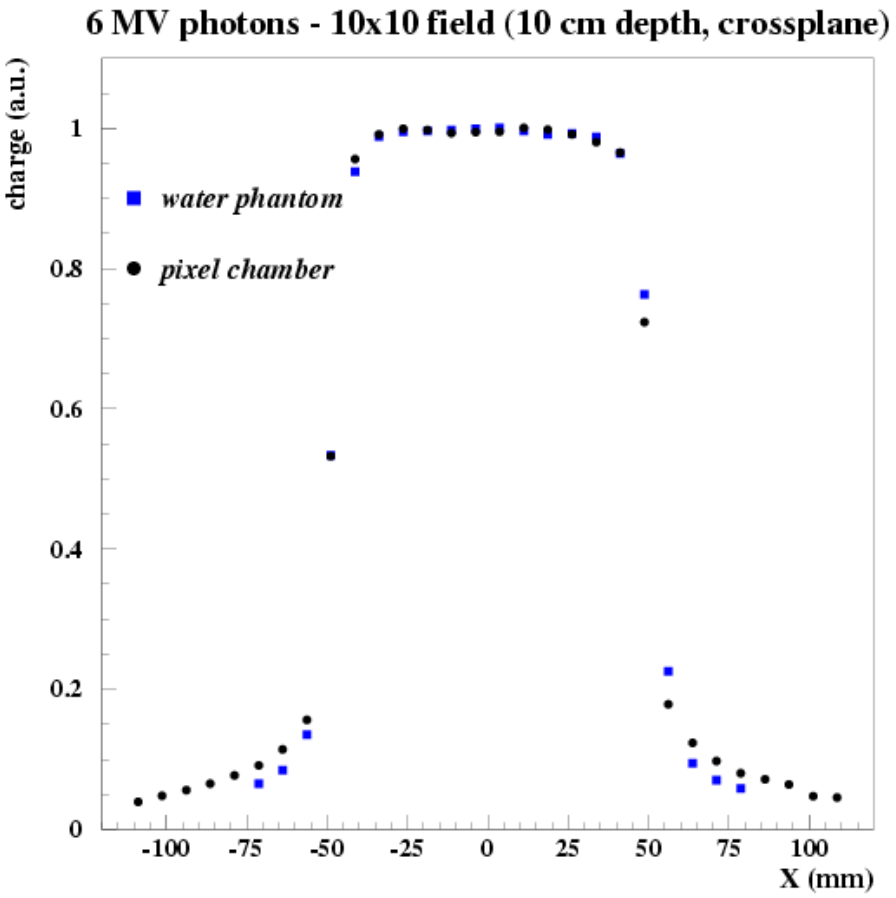


Figura 4.12: Profili dei campi (5x5) cm^2 e (10x10) cm^2, misure effettuate con fotoni da 6 MV, ad una profondità di 88 mm di plexiglass (equivalenti di 100 mm di acqua).

Capitolo 5

Risultati della simulazione

In questo capitolo vengono riportati i confronti fra i dati ottenuti dalla simulazione della camera e quelli sperimentali per varie condizioni al contorno. Viene presentata l'implementazione della simulazione della camera Farmer e il confronto con i dati ricavati da questa. Gli studi effettuati riguardano l'accordo fra i profili, soprattutto ai bordi del campo di radiazione, il profilo di dose in funzione della profondita' e in ultimo l'output factor.

Introduzione

Riprodotto l'assetto sperimentale, è stato eseguito il programma implementato per ottenere dei risultati dalla simulazione MonteCarlo confrontabili con quelli acquisiti con la camera monitor a pixel. Le simulazioni iniziali sono state finalizzate a verificare l'accordo fra i dati ottenuti e quelli sperimentali, per controllare l'affidabilità dell'implementazione del programma, e a determinare l'allargamento della penombra gaussiana del fascio utilizzato che meglio rappresentasse quello prodotto da un acceleratore terapeutico. Successivamente è stato studiato il deposito di energia nella camera in funzione della profondità, per ottenere la curva di buildup dei fotoni, e sono stati misurati i profili di dose con il gantry rotante su un piano perpendicolare al lettino del paziente. Infine sono state effettuate alcune misure sull'output factor, per quantificare la variazione di energia depositata in funzione del campo di radiazione. A questo riguardo è stata implementata anche la simulazione della camera Farmer, in due configurazioni: una più semplice utilizzando solo aria e plexiglass, una più completa, composta di anodo, catodo e un mantello di sostegno.

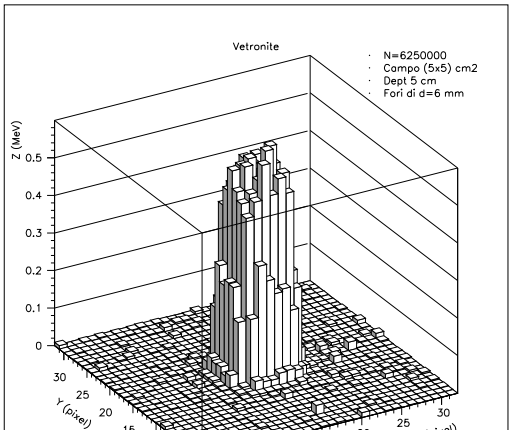
Alcuni confronti con i dati sperimentali non sono nelle identiche condizioni al contorno e possono variare alcuni parametri che non determinano però in modo decisivo l'andamento delle curve confrontate. Il problema delle simulazioni effettuate è che, per ottenere un accettabile errore statistico e soprattutto per campi grandi come quello $(20x20)\ cm^2$, il numero di eventi da utilizzare è grande e la simulazione dura diversi giorni.

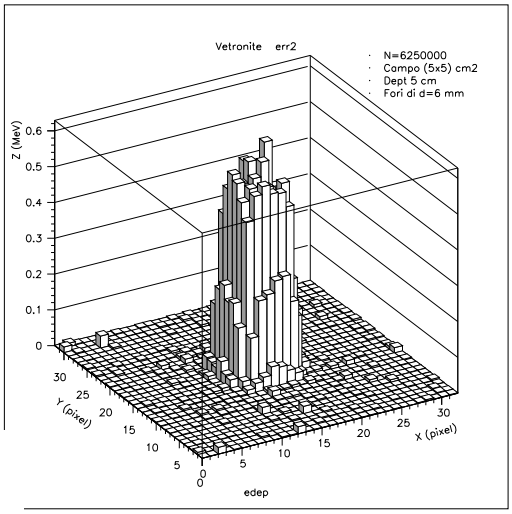
5.1 Valutazione dell'errore statistico

Ogni dato della simulazione è associato ad un errore statistico, legato al numero di eventi inviati sul rivelatore. Sono stati inizialmente eseguiti dunque alcuni run con seed iniziale diverso per ogni simulazione, con il campo $(5x5)\ cm^2$ e un numero di eventi $N = 6.25 \cdot 10^6$. La camera è ad una profondità di 5 cm di plexiglass, con la griglia di vetronite di spessore 3 mm, avente fori del diametro $d = 6\ mm$.
Dagli istogrammi ottenuti sono stati calcolati i valori delle energie depositate nei 4 e nei 16 pixel centrali, ricavando così un'energia media, $E_m^{4/16}$, che è stata considerata l'energia media del run. Si è inoltre calcolata la deviazione standard σ, che può essere interpretata come errore statistico associato ad ogni misura ottenuta dalla simulazione con N eventi. Aumentando il numero di particelle inviate sul rivelatore, l'errore non varia in modo significativo, perchè diminuisce secondo $1/\sqrt{N}$.
Riportiamo le tabelle dei dati ricavati dai quattro run eseguiti.

PRIMO RUN				
Pixel	**15**	**16**	**17**	**18**
15	0,488	0,580	0,494	0,531
16	0,501	0,530	0,604	0,508
17	0,563	0,496	0,518	0,601
18	0,557	0,569	0,539	0,497
Energia	**E4 = 0,537**		**E16 = 0,536**	

Figura 5.1: Tabella dei valori di energia, espressa in MeV depositata nei 16 pixel centrali della camera durante il primo run. Da questi vengono calcolati due valori medi, E_m^4 e E_m^{16} riportati nell'ultima riga.


Vetronite
N=6250000
Campo (5x5) cm2
Dept 5 cm
Fori di d=6 mm
Z (MeV)
0
0.1
0.2
0.3
0.4
0.5
Y (pixel)
30
25
20
25
30


Vetronite err2
N=6250000
Campo (5x5) cm2
Dept 5 cm
Fori di d=6 mm
Z (MeV)
0
0.1
0.2
0.3
0.4
0.5
0.6
Y (pixel)
30
25
20
15
10
5
0
X (pixel)
0
5
10
15
20
25
30
edep

SECONDO RUN				
Pixel	**15**	**16**	**17**	**18**
15	0,529	0,549	0,523	0,52
16	0,538	0,496	0,530	0,493
17	0,529	0,551	0,592	0,507
18	0,532	0,589	0,542	0,633
Energia	**E4 = 0,542**		**E16 = 0,541**	

Figura 5.3: Tabella dei valori di energia depositata nei 16 pixel centrali della camera durante il secondo run.

TERZO RUN				
Pixel	**15**	**16**	**17**	**18**
15	0,472	0,528	0,489	0,483
16	0,503	0,538	0,541	0,538
17	0,516	0,519	0,493	0,542
18	0,576	0,550	0,560	0,602
Energia	**E4 = 0,523**		**E16 = 0,528**	

Figura 5.4: Tabella dei valori di energia depositata nei 16 pixel centrali della camera durante il terzo run.

QUARTO RUN				
Pixel	**15**	**16**	**17**	**18**
15	0,555	0,541	0,535	0,532
16	0,516	0,557	0,557	0,495
17	0,529	0,546	0,505	0,520
18	0,543	0,575	0,562	0,528
Energia	**E4 = 0,541**		**E16 = 0,537**	

Figura 5.5: Tabella dei valori di energia depositata nei 16 pixel centrali della camera durante il quarto run.

E' stata quindi calcolata la deviazione standard dei valori medi sui quattro pixel centrali E_4 e sui 16 centrali E_{16} usando la classica relazione:

$$\sigma^2 = \frac{1}{(N-1)} \sum_{i=0}^{N} (x_i - \bar{x})^2$$

dove per x_i si intende rispettivamente E_4 ed E_{16}, e N e' il numero di run pari a 4.
L'errore statistico su E_4 ottenuto con un numero di eventi pari a $6.25 \cdot 10^6$ è $\sigma = 0.9\%$. L'errore su E_{16} ottenuto e' $\sigma = 0.5\%$, che è circa un fattore due migliore rispetto ad E_4, il che e' compatibile con il fatto che si media su un maggiore numero di pixel per un fattore quattro.

Con buona approssimazione si puo' affermare che l'errore sui valori ottenuti con le simulazioni, che prevedevano un numero di eventi pari a $6.25 \cdot 10^6$ con un campo $(5x5)$ cm^2 e a scalare per campi maggiori in ragione del rapporto delle aree, sia circa dello 1% quando la media e' stata fatto su quattro pixel, circa dell 0.5 % per medie su 16 pixels, e infine del 2% qualora si volesse considerare soltanto il pixel centrale.
Ricordiamo, esplicitamente, il numero di eventi simulato in ragione del campo:

- $N_5 = N = 6.25 \cdot 10^6$ fotoni per il campo $(5x5)$ cm^2;
- $N_{10} = 4N = 2.5 \cdot 10^7$ fotoni per il campo $(10x10)$ cm^2;
- $N_{15} = 9N = 5.625 \cdot 10^7$ fotoni per il campo $(15x15)$ cm^2;
- $N_{20} = 16N = 10^9$ fotoni per il campo $(20x20)$ cm^2.

La simulazione del campo $(20x20)cm^2$ è stata in alcuni casi omessa per problemi di tempo di simulazione. Vi e' poi una categoria di simulazioni per le quali non si richiedeva un confronto in funzione del campo, come la curva di buildup, o i profili del gantry rotante e simili, nel qual caso si sono generati 10^7 eventi. Con buona approssimazione, anche in questo caso, assumeremo che l'errore statistico sia dello 1%.

5.2 Profilo gaussiano

Per misurare quanto l'allargamento della penombra che circonda il fascio influenzasse l'andamento delle code dei profili di dose ottenuti dalla simulazione, sono stati effettuati dei run con $\sigma = 2.5mm$ e $\sigma = 5mm$, dove con σ si indica la semilarghezza a metà altezza della gaussiana che descrive i bordi del fascio.

Le condizioni della simulazione sono:

- $N = 10^7$ di fotoni;
- $dept = 9\ mm$ di profondità;
- $C = (5x5)\ cm^2$ per il campo di radiazione;
- $Gap = 3\ mm$ di spessore;
- $Fori = 6\ mm$ di diametro.

Sono stati ricavati gli istogrammi (5.7) e (5.8) relativi alle due simulazioni con differenti valori di σ.
In ogni istogramma sono state sommate le quattro slice centrali, corrispondenti alle file di pixel 15, 16, 17 e 18[1], ed è stato confrontato il profilo così ottenuto con quello sperimentale, per verificare quale condizione simulasse meglio il fascio prodotto dall'acceleratore terapeutico utilizzato per l'acquisizione dati.
Dagli istogrammi ottenuti dalle simulazioni si sono ricavati i dati riportati nella tabella (5.6).

Dal confronto si può affermare che la larghezza del profilo gaussiano del fascio non influenza in modo determinante la penombra presente nelle code. I dati sperimentali sono leggermente più alti prima del dodicesimo pixel e dopo il ventunesimo. Un migliore accordo in queste zone fra simulazione e dati sperimentali si ha con un valore di σ pari a 5 mm. Questo valore è dunque stato usato per le simulazioni successive. Notiamo che nei pixel al centro della camera, la distribuzione di energia depositata nella simulazione non è uniforme ed è presente una visibile

[1]Si ricorda che la camera è una matrice (32x32), quindi le file di pixel attorno a 16 sono quelle centrali.

SIGMA		
pixel	**2,5 mm**	**5 mm**
10	0	0,038
11	0	0,031
12	0,051	0,182
13	1,096	1,461
14	2,788	2,919
15	2,802	2,830
16	2,802	2,954
17	2,891	2,892
18	2,932	2,850
19	2,822	2,692
20	1,225	1,427
21	0,051	0,134
22	0	0,024
23	0,024	0

Figura 5.6: Valori di energia depositata, in unità arbitrarie, nei 14 pixel centrali della camera, sommando quattro slice. L'energia depositata nei primi e negli ultimi 9 pixel è trascurabile.

fluttuazione statistica. Per ottenere un miglior accordo con il profilo sperimentale si dovrebbe aumentare il numero di eventi utilizzato, compatibilmente con il tempo di elaborazione dei dati. Ogni differenza fra dati sperimentali e simulati rientra però nel margine di errore associato alle misure, pari allo 0.9%.

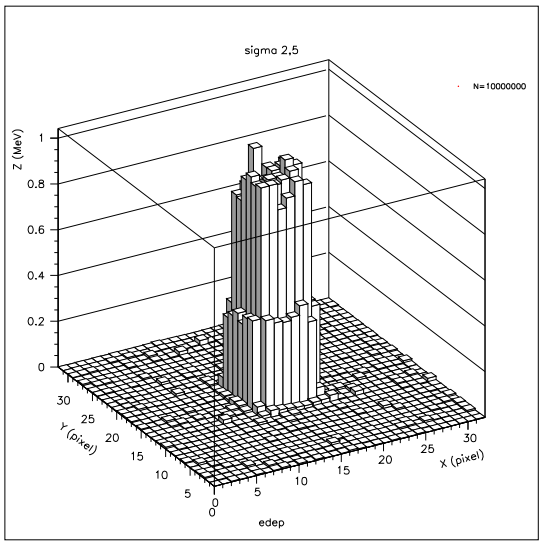


Figura 5.7: Istogramma relativo al run con $\sigma = 2.5\ mm$.

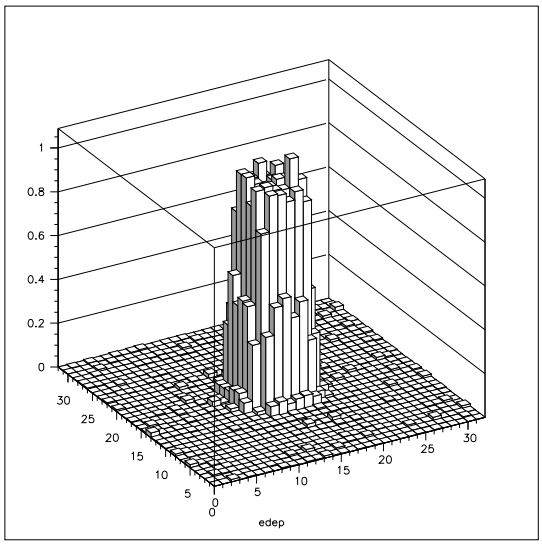


Figura 5.8: Istogramma relativo al run con $\sigma = 5\ mm$.

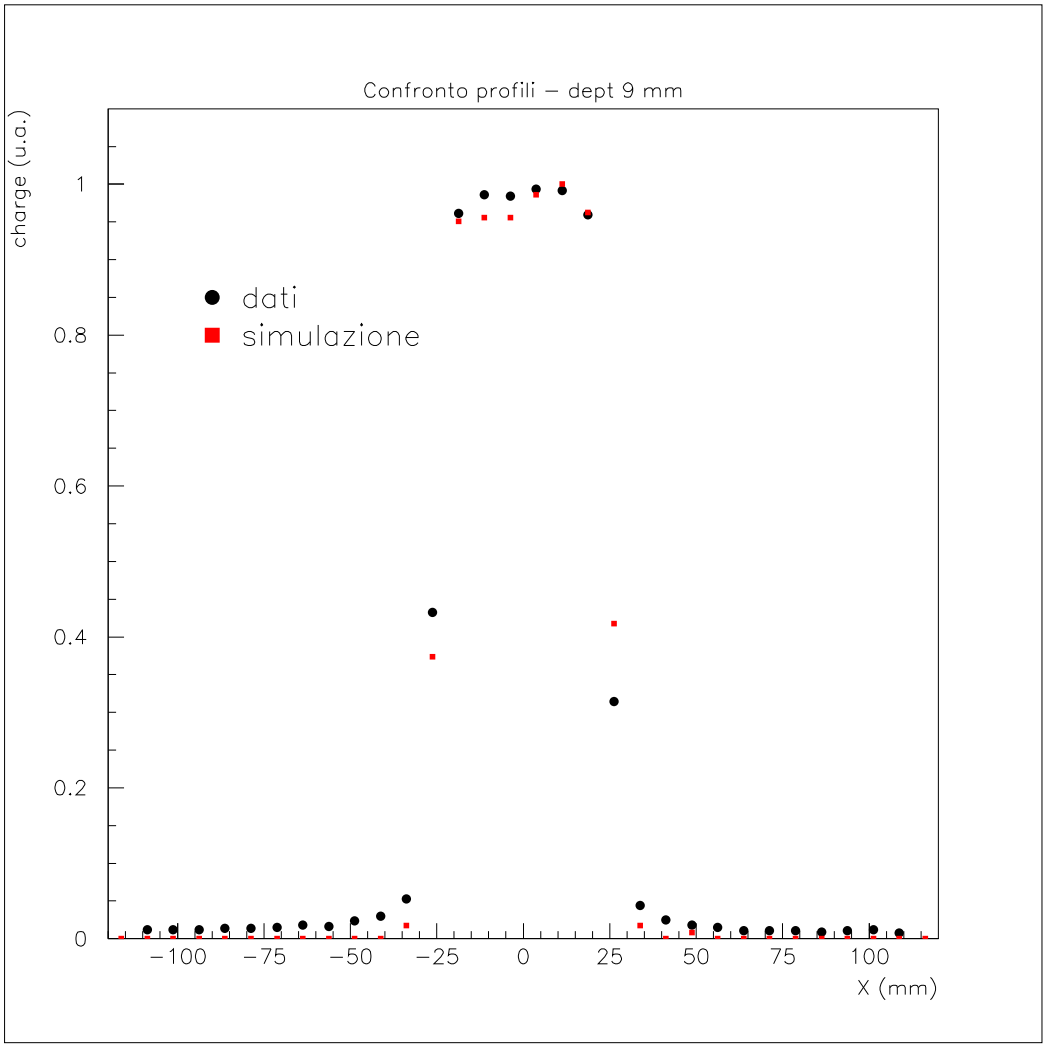


Figura 5.9: Confronto tra il profilo di energia depositata, per un campo di radiazione $(5x5)\ cm^2$, ottenuto dalla simulazione, con semilarghezza della gaussiana del fascio pari a 2.5 mm, e quello sperimentale.

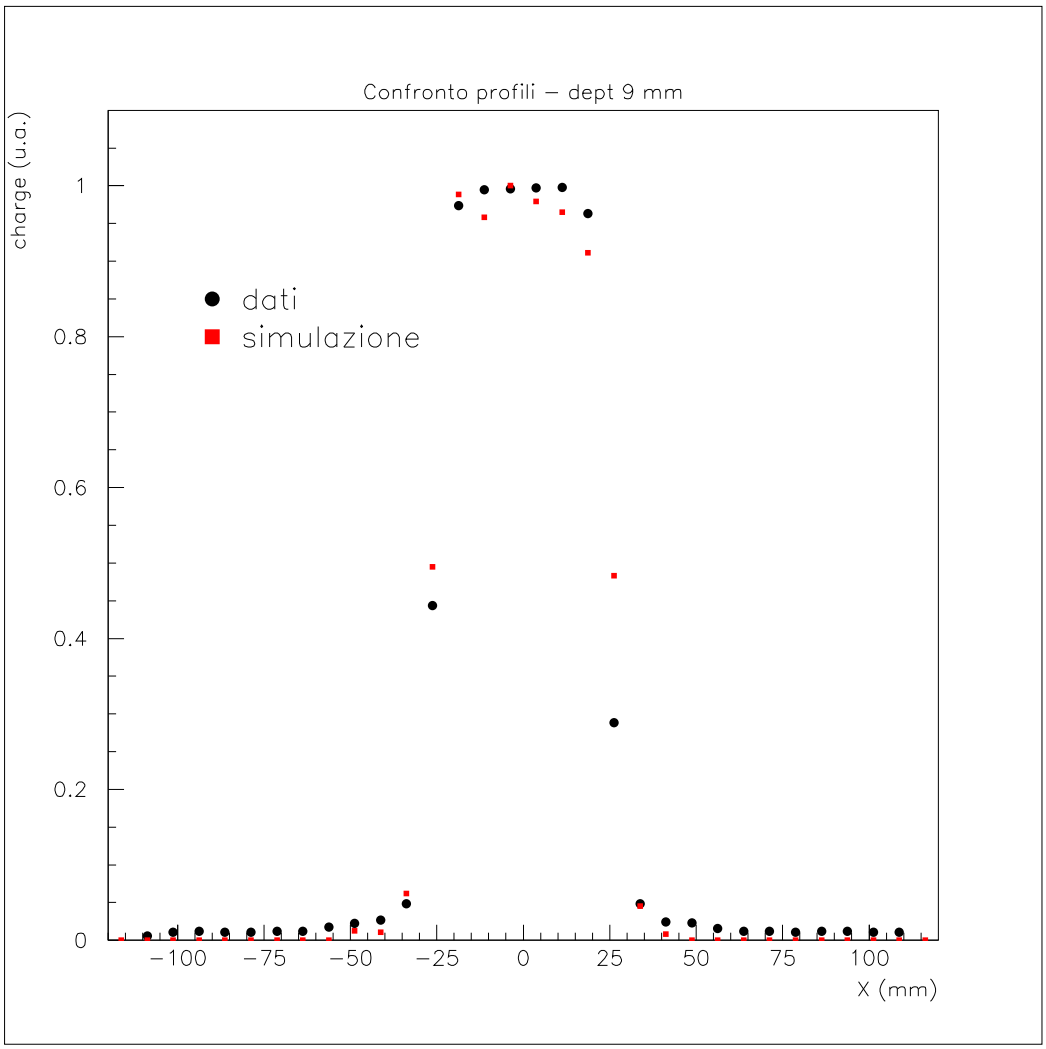


Figura 5.10: Confronto fra i profili della simulazione, con $\sigma = 5\ mm$, e dei dati sperimentali, per un campo di radiazione di $(5x5)\ cm^2$.

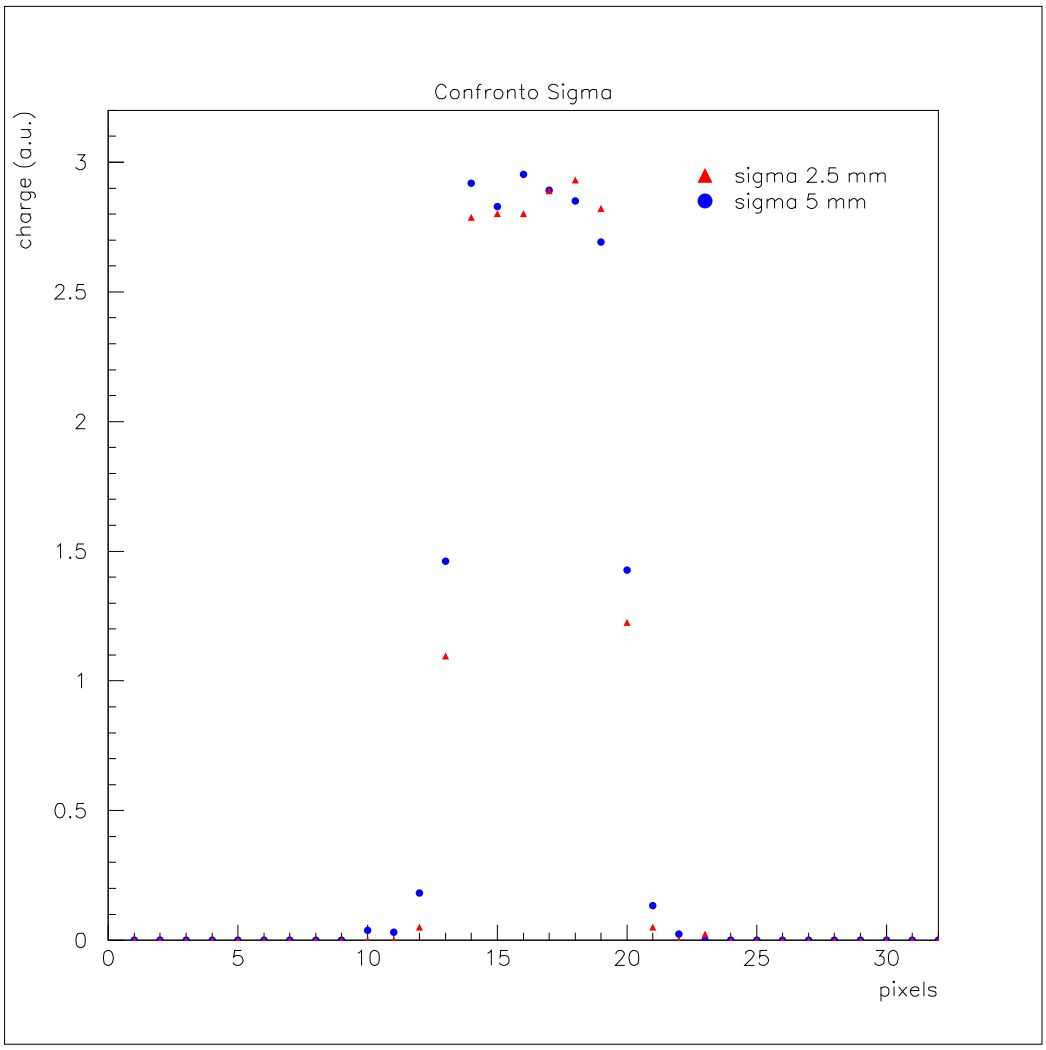


Figura 5.11: Confronto fra i profili della simulazione con i due valori di σ, per un campo di radiazione di $(5x5)\ cm^2$.

5.3 Simulazione della dose in funzione della profondità

La risposta delle camere a ionizzazione e in generale di strumenti di misura di dose deve essere verificata in funzione della profondità e controllata paragonandola a quella di dosimetri standard. Man mano che si aggiunge spessore davanti alla camera l'energia del fascio si degrada e la corretta [2] risposta indica che la camera in esame non presenta anomalie in funzione dell'energia. Per questo la camera a pixel e' stata studiata in funzione dello spessore di materiale (plexiglass) messo davanti e confrontata con le misure effettuate con la camera Farmer nelle identiche condizioni.0 Per lo stesso motivo e' importante verificare l'accordo nella posizione del buildup. Qui si ricorda che con il termine buildup si intende l'effetto per cui il deposito di energia cresce in funzione della profondita' nei primi millimetri per poi diminuire quasi esponenzialmente.

I dati sperimentali per la camera Farmer e quella a pixel hanno mostrato un ottimo accordo e quindi si e' voluto verificare anche la correttezza della simulazione in funzione dell'energia del fascio. In altri termini e' importante controllare che la *risposta* della simulazione in funzione dell'energia segua fedelmente le misure sperimentali. Sono stati eseguiti alcuni run con diversi spessori di plexiglass davanti alla camera per valutare la distribuzione dell'energia depositata in funzione della profondità.
Le condizioni della simulazione sono:

- $N = 10^7$ per il numero di fotoni;
- $\sigma = 5\ mm$ per il profilo del fascio;
- $C = (10x10)\ cm^2$ per il campo di radiazione;
- $Gap = 5\ mm$ di spessore;
- $Lastraforata$ di plexiglass;
- $Fori = 6\ mm$ di diametro.

[2] Qui con il termine *corretto* si intende di fatto l'accordo con uno dei dosimetri standard e accettati.

I valori di deposito di energia, normalizzati al buildup[3], ottenuti con la simulazione sono riportati nella tabella (5.12) per le configurazioni della lastra di plexiglass e di vetronite rispettivamente. Con la lastra di vetronite, la simulazione e' stata fatta per un campo $(5x5)$ cm^2: per questa ragione non e' direttamente confrontabile con i dati sperimentali. Questi risultati sono stati confrontati con quelli sperimentali della camera Farmer e della camera monitor a pixel come mostrato in Figura (5.13). La normalizzazione delle curve per quanto riguarda l'energia depositata e' stata fatta al punto di massimo deposito di energia. Dalla figura (5.13) si rileva che i risultati della simulazione non sono in accordo con i risultati sperimentali. In particolare la simulazione tende a sovrastimare il deposito di energia in funzione della profondita'. I motivi della discrepanza non sono facilmente intuibili e sara' necessaria un'ulteriore investigazione. In figura (5.14) vengono riportati i risultati del MonteCarlo ottenuti nei primi spessori di plexiglass: la camera ha la lastra di vetronite e il campo utilizzato e' di $(5x5)$ cm^2.

Dept Plex (mm)	Dept H2O (mm)	Plex (u.a.)	Vetronite (u.a.)
9	10,225	1	1
50	56,803	0,881	0,894
100	113,605	0,681	0,695
150	170,408	0,598	0,563
200	227,211	0,491	0,474

Figura 5.12: Tabella dei valori medi di energia depositata nei quattro pixel centrali della camera a diverse profondità, normalizzata a 9 mm. I valori del plexiglass sono relativi ad un campo di $(10x10)$ cm^2, quelli della vetronite ad un campo $(10x10)$ cm^2

Dal confronto fra i dati della simulazione e quelli sperimentali si può concludere che il codice Geant4, per particelle a bassa energia, sovrastima il segnale prodotto nel gap sensibile, sia nel caso di una lastra di vetronite, sia per la lastra in plexiglass. Questo può essere spiegato con il fatto che Geant4 è nato per simulazioni ad alte energia, e solo in un secondo tempo si è iniziato ad utilizzarlo anche per particelle meno energetiche, come quelle in uso in campo medico. Le sezioni d'urto implementate nel toolkit che permettono di calcolare con precisione alte dosi assorbite, possono dunque essere meno precise quando il range di energia considerato è notevolmente più basso.

[3]Il buildup per un campo $(10x10)$ cm^2 e un fascio di fotoni a 6 MV si trova ad una profondita' di circa 10 mm di acqua equivalente.

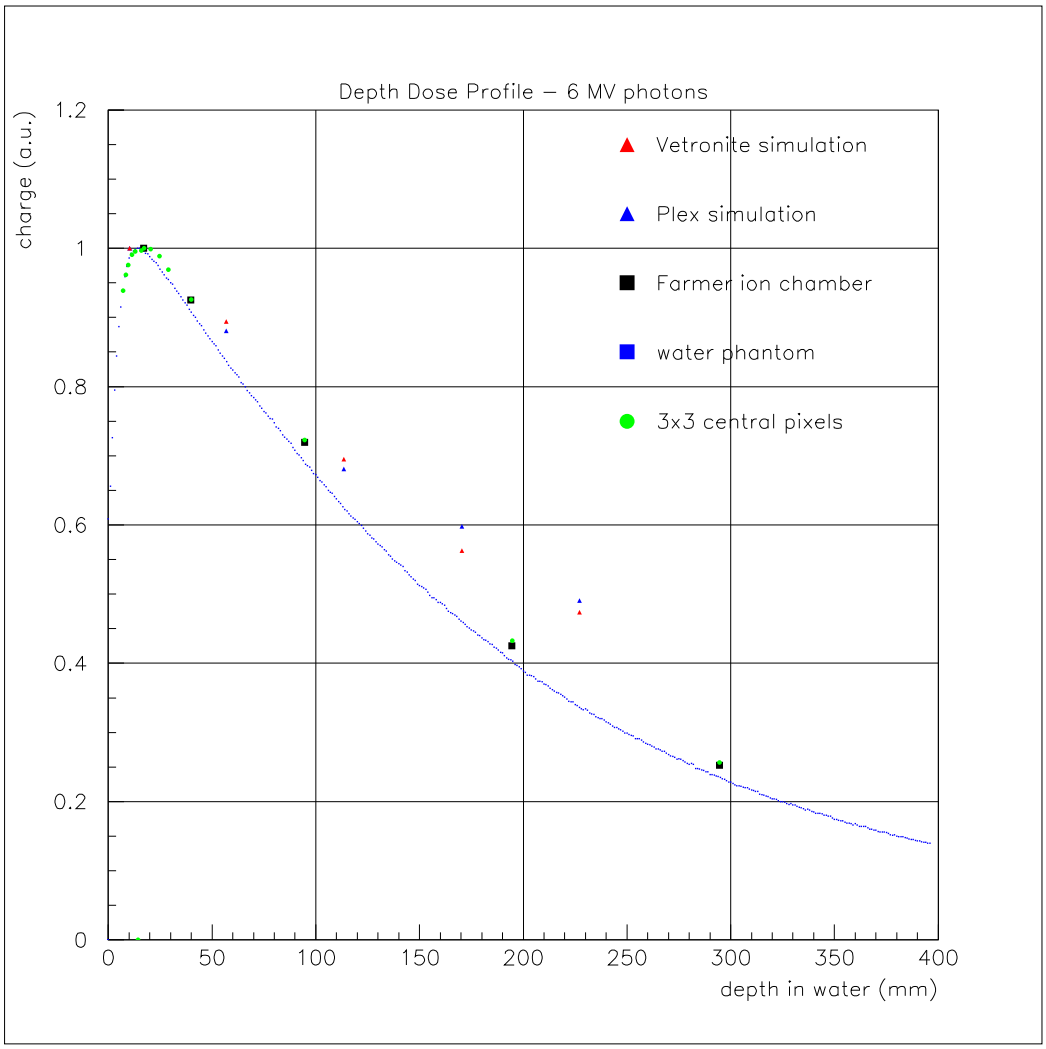


Figura 5.13: Curva di buildup ottenuta con una camera standard immersa nel fantoccio ad acqua (water phantom), con la quale confrontiamo i dati sperimentali della Farmer e della camera monitor

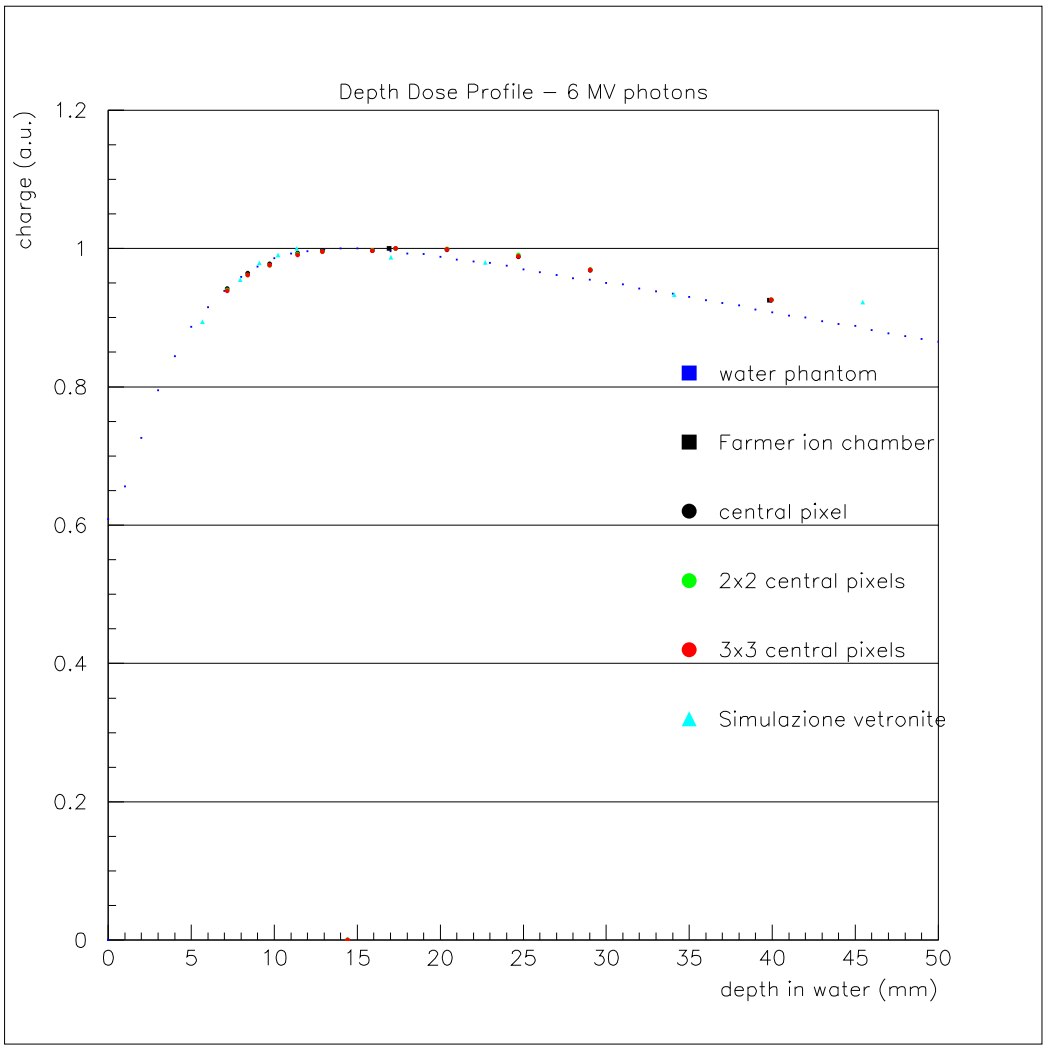


Figura 5.14: Ingrandimento della zona di buildup. Si notano le differenze fra le due curve della camera (in rosso i valori sperimentali, in azzurro la simulazione) e quella del fantoccio ad acqua (in blu). La simulazione con la lastra di vetronite nei primi spessori è relativa ad un campo $(5x5)$ cm^2. Possiamo concludere che fino ad uno spessore di 4 cm la simulazione rappresenta bene i risultati, a spessori maggiori invece sovrastima la dose assorbita.

5.4 Gantry rotante

Per irradiare il paziente con diversi campi viene usato un gantry rotante. Questo accorgimento evita di dover cambiare la posizione del paziente e quindi di velocizzare i trattamenti. Sono state eseguite delle misure, dunque, con la camera monitor a pixel per angoli di $10^0, 20^0$ e 30^0, per verificare i valori di energia depositata anche nel caso di non perpendicolarità del fascio. Sono stati effettuati dei run per verificare l'accordo dei dati sperimentali con la simulazione e in particolare sono stati confrontati i profili.
Inizialmente si è simulato un fascio perpendicolare (0^0), e il profilo ottenuto è stato confrontato con quello sperimentale. Il procedimento è stato ripetuto per fasci ad angoli pari a $10^0, 20^0$ e 30^0.

Le condizioni della simulazione sono:

- $N = 10^7$ per il numero di fotoni;
- $\sigma = 5\ mm$ per il profilo del fascio;
- $C = (5x5)\ cm^2$ riguardo al campo di radiazione;
- $dept = 9\ mm$ la profondità in plexiglass;
- $Gap = 3\ mm$ di spessore;
- $Fori = 6\ mm$ di diametro.

Nella tabella (5.15) si riportano i valori ottenuti dalla simulazione dell'energia depositata in funzione del numero di pixel. Per limitare le fluttuazioni di natura statistica si sono sommate quattro righe di pixel prese attorno al centro della camera. Esplicitamente il valore riportato in corrispondenza del pixel 16, per esempio, e' la somma sui pixel (15,16), (16,16), (17,16), e (18,16) e cosi' via.
Concludiamo dai risultati della simulazione che, con un fascio inclinato di un angolo α rispetto alla normale alla camera, la distribuzione di energia depositata nei pixel si sposta dalla parte del fascio. Il fenomeno è molto più evidente nella simulazione che nei dati sperimentali.

pixel	ang 0	ang 10	ang 20	ang 30
10	0	0,032	0	0
11	0,031	0,032	0	0,044
12	0,132	0,160	0,199	0,270
13	1,413	0,387	1,522	1,679
14	2,876	2,828	2,983	3,153
15	2,910	3,012	3,106	3,459
16	2,801	3,0405	2,976	3,314
17	2,890	2,860	3,041	3,201
18	2,828	2,856	2,918	3,169
19	2,727	2,856	2,968	3,266
20	1,507	1,437	1,349	1,389
21	0,159	0,153	0,141	0,165
22	0	0,018	0	0

Figura 5.15: Valori di energia depositata, in unità arbitrarie, nei 13 pixel centrali della camera, dopo che sono state sommate le quattro slice centrali.

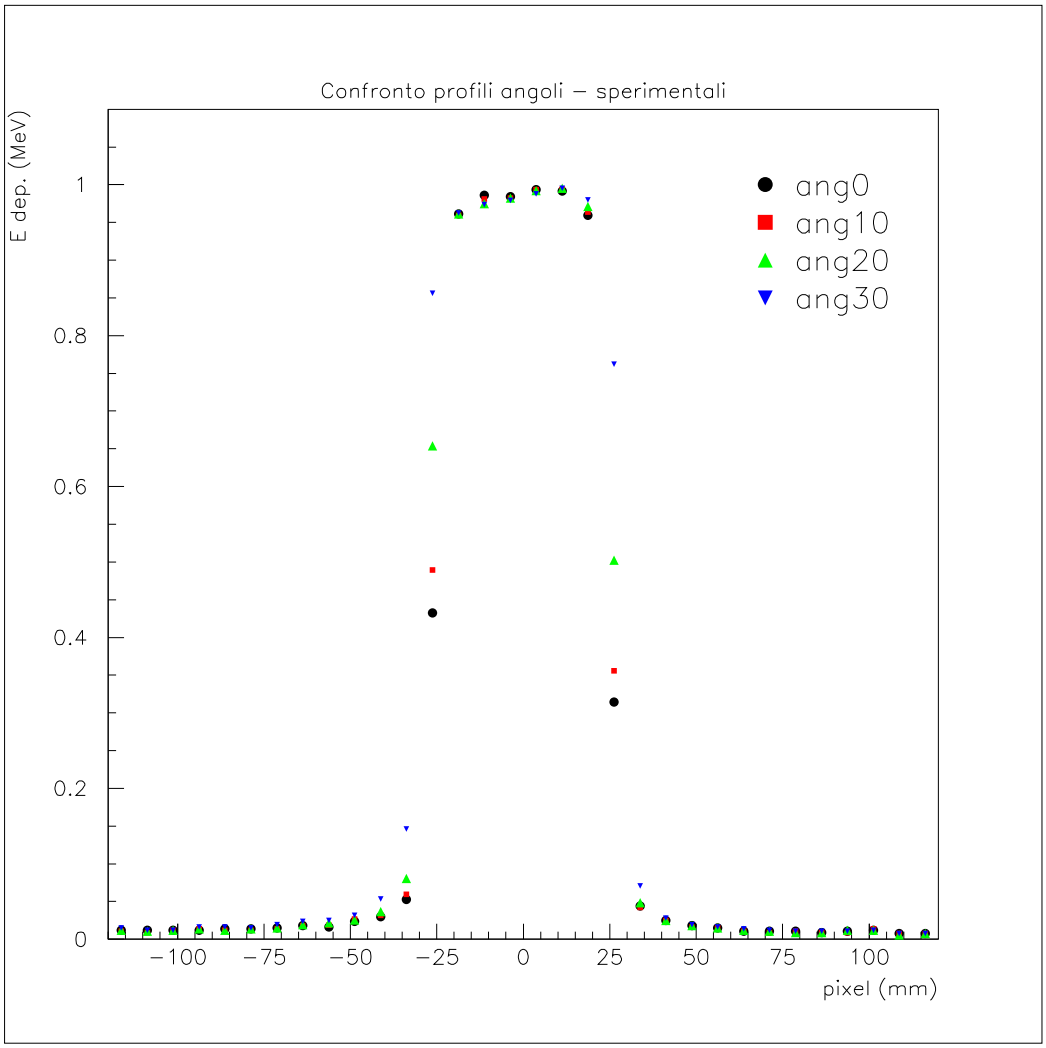


Figura 5.16: Confronto fra i profili ottenuti con i dati sperimentali variando l'angolo del fascio a step di 10^0, da 0^0 a 30^0.

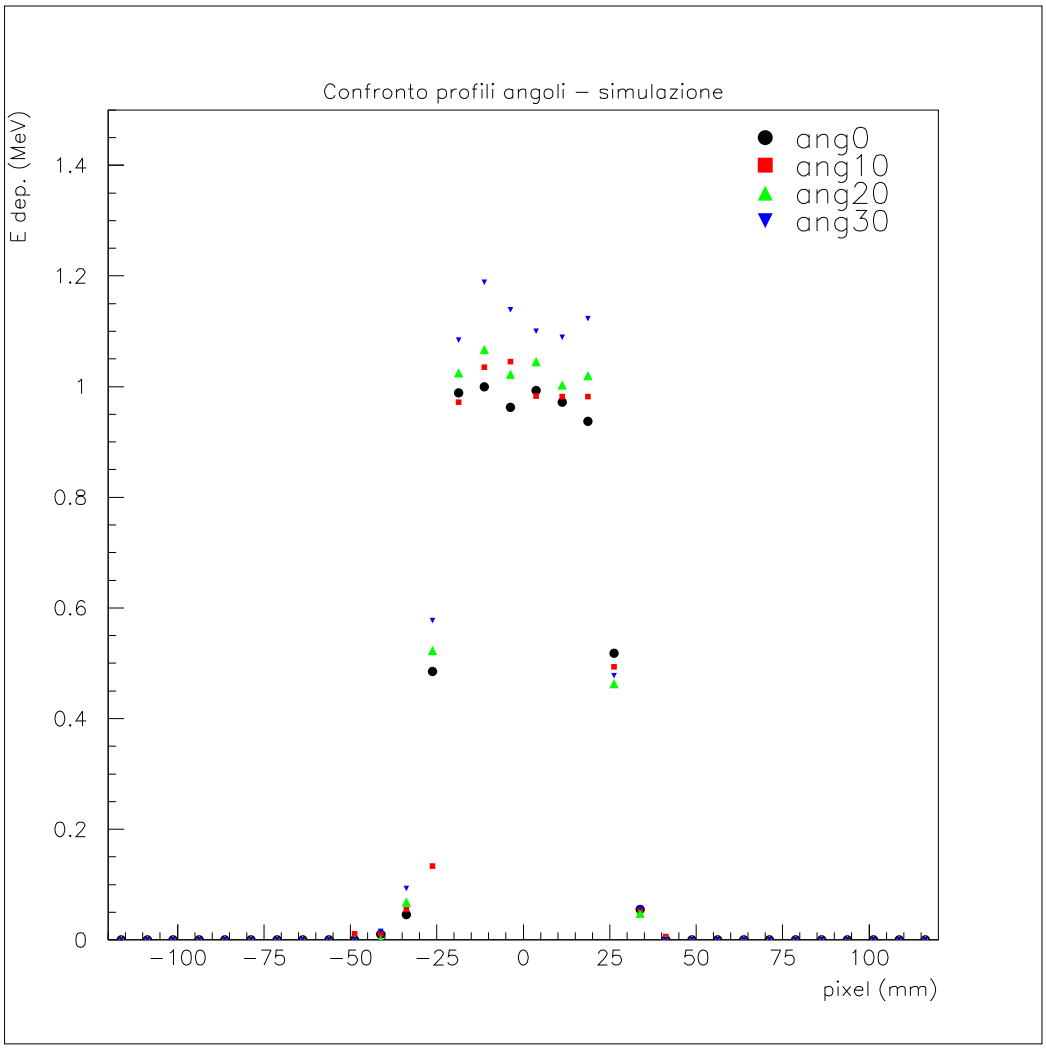


Figura 5.17: Confronto fra i profili ottenuti con la simulazione variando l'angolo del fascio a step di 10^0, da 0^0 a 30^0. Sono state sommate le quattro slice centrali, dalla 15 alla 19.

5.5 Confronto dell'Output Factor

Un parametro importante che entra nella stesura dei piani di trattamento e' il cosidetto output factor (OF). Si consideri la dose D_c rilasciata al centro di un campo quadrato, si avra' che D_c aumenta in funzione delle dimensioni del campo. Ricordiamo che indipendentemente dal campo il numero di fotoni che *escono* dalla testata dell'acceleratore e' costante e viene misurato dalla camera installata nella testata stessa. Sono i collimatori a determinare le dimensioni del campo. L'effetto e' dovuto principalmente al contributo dei fotoni per lo piu' di bassa energia che dai bordi del campo vanno ad irradiare la parte centrale dello stesso. Aumentando le dimensioni del campo aumentano altresi' i contributi dovuti ai fotoni che vanno a sovrapporsi.

Si rileva inoltre che la misura standard di OF e' fatta alla profondita' di 10 cm di Acqua. Le simulazioni e i dati sperimentali si intenderanno relativi alle suddette condizioni, a meno che non sia esplicitamente indicato.

Ripercorrendo storicamente le condizioni sperimentali si sono ottenuti qualitativamente risultati con la camera a pixel con gap di aria in disaccordo rispetto a quelli camera Farmer, che ovviamente costituiscono lo standard. Si sostitui' al gap d'aria una lastra forata di vetronite. I $32x32$ fori di diametro pari a 6 mm erano posti in corrispondenza ai pixel. Si pensava che la discrepanza, nel caso di un gap di aria, fosse dovuta interamente all'effetto di particelle di bassa energia e tipicamente a largo angolo che potevano percorrere in aria distanze anche relativamente grandi. I dati sperimentali ottenuti con la camera a pixel e lastra di vetronite era ancora in disaccordo.

Lo scopo della presente simulazione e' stato quello di studiare le possibile cause e quindi individuare gli accorgimenti per curare la divergenza. Si e' investigato sia la scelta del materiale sia la dimensione dei fori della lastra. Per quantificare l'entita' della discrepanza fra camera a pixel con lastra forata di vetronite e camera Farmer si mostrano in figura 5.19 le due curve sperimentali normalizzate entrambe al campo $(10x10)\ cm^2$.

Il problema della corretta normalizzazione nel caso del MonteCarlo non e' facilmente risolvibile. Un metodo sarebbe stato quello di simulare i fotoni gia' a

partire dalla testata stessa. Quindi seguirli attraverso i collimatori per arrivare finalmente al fantoccio di plexiglass.
Questo metodo presentava due problemi:

a) non era nota la geometria della testata e dei collimatori;
b) i tempi di simulazione sarebbero stati troppo lumghi.

Si e' invece scelto di far partire la simulazione all'uscita dei collimatori. Questa scelta presenta l'indubbio vantaggio di rendere piu' veloce la simulazione. D'altro canto richiede di scalare correttamente il numero di fotoni in funzione del campo. Inoltre la distribuzione spaziale dei fotoni sui bordi del campo non e' nota sperimentalmente. Questo problema e' stato risolto studiando le variazioni del profilo della dose rilasciata in funzione dell'ampiezza delle gaussiane di raccordo (vedere il paragrafo 2 di questo capitolo).

Il numero di fotoni in funzione del campo e' stato fatto variare in accordo con l'area, scelta in prima approssimazione corretta. Si e' dovuto valutare effetti secondari di normalizzazione imponendo l'accordo dell'output factor simulato e misurato per la camera Farmer.

Esplicitamente i valori dell'output factor sono calcolati secondo la formula seguente e riportati nella tabella (5.18) dove si confrontano i valori relativi alla camera Farmer e alla camera a pixel:

$$OF_n = \frac{Q_n}{Q_{10}} = \frac{n^o\ conteggi/carica\ nel\ campo\ considerato}{n^o\ conteggi/carica\ nel\ campo\ (10x10)\ cm^2}$$

La curva della carica raccolta, in funzione del campo di radiazione, relativa alla camera monitor a pixel ha una maggior pendenza rispetto a quella della Farmer. Questo indica che gli effetti dovuti a particella di bassa energia e largo angolo sono maggiori nella camera a pixel.

O F	Farmer	Camera
Campo (5x5)cm2	0,896	0,853
Campo (10x10)cm2	1	1
Campo (15x15)cm2	1,058	1,114
Campo (20x20)cm2	1,097	1,201

Figura 5.18: Valori sperimentali dell'output factor della camera Farmer e della camera monitor a pixel.

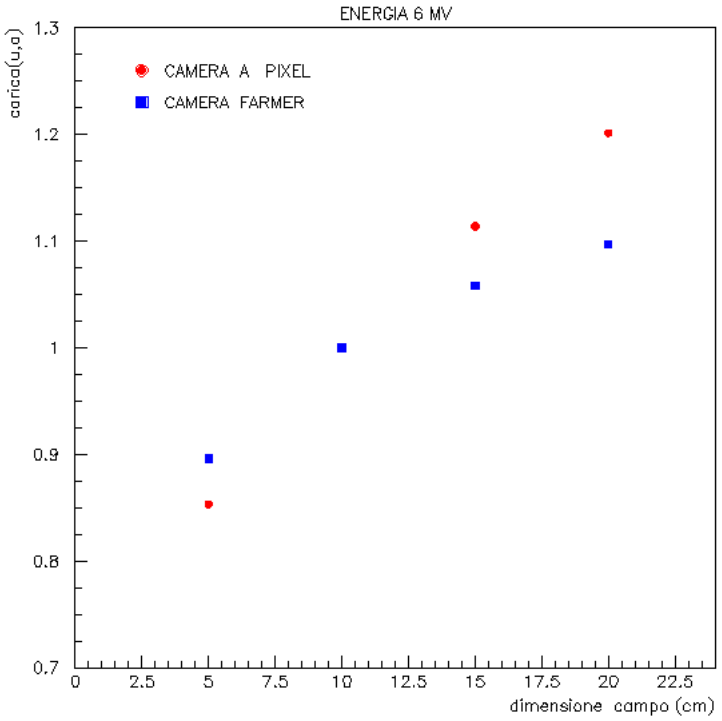


Figura 5.19: Confronto fra la carica depositata nei campi $(5x5), (10x10), (15x15) e (20x20)$ cm^2, nella camera Farmer, presa come modello, e nella camera monitor a pixel. Le due curve sono state normalizzate entrambe a 1 rispetto al campo 10.

5.5.1 Simulazione della camera Farmer

La camera Farmer simulata è una camera a ionizzazione a sezione cilindrica, modello IC10, con un volume sensibile pari a 0.14 cm^3. E' costituita da un cilindretto di aria di raggio 3 mm e altezza 3.3 mm, terminante con una calotta sferica di ugual raggio. Al centro della cameretta è presente un anodo di alluminio, di diametro 1 mm, sulle pareti si trova invece il catodo di grafite, di spessore 0.5 mm. La cameretta è sorretta da un mantello esterno di plexiglass, a sezione cilindrica, spesso 1.5 cm. La camera Farmer viene inserita in un cubo di plexiglass, alla profondità voluta, e posta al centro del campo di radiazione.

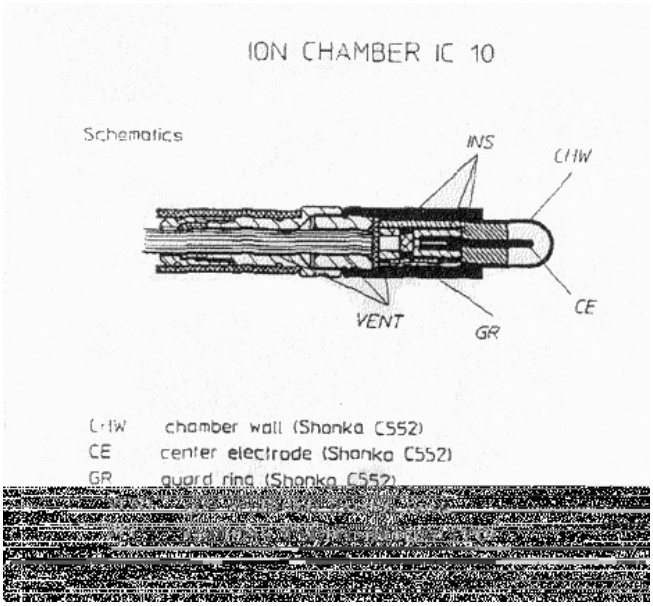


Figura 5.20: Rappresentazione schematica della camera Farmer utilizzata.

Di questa camera sono state effettuate due simulazioni: una semplificata, con un cilindro e una semisfera contenenti solo aria e immersi in plexiglass, l'altra completa di anodo, catodo e mantello. Le due configurazioni hanno identica geometria e si differenziano per il tipo di materiale che costituisce le pareti della camera. La geometria è stata descritta nell'omonima classe, attraverso la definizione di un cilindro e di una semisfera che vengono uniti attraverso un'operazione

booleana, posizionati sugli assi x e y al centro di un cubo di lato 20 cm, ad una profondità fissata. Il fascio utilizzato e i processi fisici presi in considerazione sono gli stessi della simulazione della camera, poichè il tipo di particelle incidenti e le interazioni di maggior importanza non variano.

Farmer semplificata

Nella prima simulazione vengono definiti due elementi geometrici che formano il volume sensibile, un cilindro e una semisfera. Il solido di unione, centrato inizialmente sull'asse z, viene poi ruotato di un angolo ϕ attorno all'asse x.

Segue il codice commentato utilizzato per la loro implementazione:
// viene assegnato il valore di default alle variabili

```
Geometria::Geometria()
{
  // --- World
  WorldSizeZ = 2.5*m;
  WorldSizeXY = 2.5*m;

  // --- Cubo
  CubePosZ = 0.0*mm;
  CubeSizeZ = 20.0*cm;
  CubeSizeXY = 20.0*cm;

  // --- Cilindro
  CyliInnerRadius = 0.0*mm;
  CyliOuterRadius = 2.6*mm;
  CyliStartAngle = 0.0*deg;
  CyliSpanningAngle = 360.0*deg;
  CyliHeight = 3.3*mm;
  CyliPosZ = -(CubeSizeZ/2 - Dept);
  CyliPosX = 0.0*mm;
  CyliPosY = 0.0*mm;

  // --- Sfera
  SferaInnerRadius = 0.0*mm;
```

```
  SferaOuterRadius =  CyliOuterRadius;
  SferaStartAngle = 0.0*deg;
  SferaSpanningAngle = 360.*deg;
  SferaStartAper = 0.*deg;
  SferaSpanningAper = 90.*deg;

  Dept = 83.35*mm;  //  profondità
  phi = 270.*deg;   //  ang. di rotazione
  ...
}
```

// in questo metodo si definiscono e si posizionano i solidi

```
G4VPhysicalVolume* Geometria::ConstructFarmer()
{
  ...

// --- World
solidWorld = new G4Box("World",
                       WorldSizeXY/2,
                       WorldSizeXY/2,
                       WorldSizeZ/2);

logicWorld = new G4LogicalVolume(solidWorld,
                                 defaultMaterial,
                                 "World");
logicWorld->SetVisAttributes(G4VisAttributes::Invisible);

phyWorld = new G4PVPlacement(0,
                             G4ThreeVector(),
                             logicWorld,
                             "World",
                             NULL,
                             false,
                             0);
```

```
// --- Cube
solidCube = new G4Box("boxCube",
                      CubeSizeXY/2,
                      CubeSizeXY/2,
                      CubeSizeZ/2);

logicCube = new G4LogicalVolume(solidCube,
                                CubeMaterial,
                                "logCube");
logicCube->SetVisAttributes(G4VisAttributes::Invisible);

phyCube = new G4PVPlacement(0,
                            G4ThreeVector(0,0,CubePosZ),
                            logicCube,
                            "Cube",
                            logicWorld,
                            false,
                            0);

// --- Cylinder  &  Sfera
// viene solo costruito il volume solido,
// necessario all'operazione booleana
solidCylinder = new G4Tubs("Cylinder",
                            CyliInnerRadius,CyliOuterRadius,
                            CyliHeight/2,
                            CyliStartAngle,CyliSpanningAngle);

solidSfera = new G4Sphere("Sfera",
                          SferaInnerRadius,SferaOuterRadius,
                          SferaStartAngle,SferaSpanningAngle,
                          SferaStartAper,SferaSpanningAper);

// --- Unione solidi
solidCyli = new G4UnionSolid("Cyli",
          solidCylinder,solidSfera,
```

```
          0,                                  // nessuna rotazione
          G4ThreeVector(0,0,CyliHeight/2)); // trasla la sfera di h/2

logicCyli = new G4LogicalVolume(solidCyli,
                                CyliMaterial,
                                "logCyli");

// si ruota il solido di 270 gradi
G4RotationMatrix rot;              // crea l'istanza
rot.rotateX(phi);                  // applica la rotazione su x

phyCyli = new G4PVPlacement(G4Transform3D(rot,      // rotazione
                          G4ThreeVector(CyliPosX,
                                        CyliPosY,
                                        CyliPosZ)), // posizionamento
                                        logicCyli,  // vol. logico
                                        "Cyli",     // nome
                                        logicCube,  // vol. madre
                                        false,      // op.booleane
                                        0);         // copy number

// camera colorata di azzurro
G4VisAttributes* CyliVisAttr = new
                 G4VisAttributes(G4Colour(0.5,3.0,1.0));
logicCyli->SetVisAttributes(CyliVisAttr);
...
}
```

Farmer completa

Alla simulazione precedente sono stati aggiunti alcuni elementi geometrici ed alcuni materiali per descrivere con maggior precisione la camera Farmer utilizzata. E' stato implementato il filo di anodo in alluminio, il catodo rappresentato da uno stato di grafite depositato sulle pareti del cilindro, e il mantello in plexiglass che racchiude l'intera camera. Le aggiunte alla simulazione precedente sono state descritte dal seguente codice:

// si assegnano i valori di default

```
Geometria::Geometria()
{
  ...
  // --- Anodo
  AnodoInnerRadius   = 0.0*mm;
  AnodoOuterRadius   = 1.0*mm;
  AnodoStartAngle    = 0.0*deg;
  AnodoSpanningAngle = 360.0*deg;

  // --- Catodo
  CatodoInnerRadius   = CyliOuterRadius;
  CatodoOuterRadius   = CyliOuterRadius + CatodoThickness;
  CatodoStartAngle    = 0.0*deg;
  CatodoSpanningAngle = 360.0*deg;
  CatodoThickness     = 0.5*mm;

  // --- Mantello
  MantelloInnerRadius   = CatodoOuterRadius;
  MantelloOuterRadius   = CatodoOuterRadius + MantelloThickness;
  MantelloStartAngle    = 0.0*deg;
  MantelloSpanningAngle = 360.0*deg;
  MantelloThickness     = 1.5*cm;
  ...
}
```

// costruisce gli elementi geometrici

```
G4VPhysicalVolume* Geometria::ConstructFarmer()
{
...
// --- Anodo
solidAnodo = new G4Tubs("tubsAnodo",
              AnodoInnerRadius,AnodoOuterRadius,
              AnodoHeight/2,
```

```
              AnodoStartAngle,AnodoSpanningAngle);

logicAnodo = new G4LogicalVolume(solidAnodo,
                                 AnodoMaterial,
                                 "logAnodo");

phyAnodo = new G4PVPlacement(G4Transform3D(rot,   // ruota risp. al cubo
                         G4ThreeVector(CyliPosX,
                                       CyliPosY,
                                       CyliPosZ)), // centrato
                                       logicAnodo, // vol. logico
                                       "Anodo",    // nome
                                       logicCube,  // volume madre
                                       false,      // no oper. booleane
                                       0);         // copy number

// filo di anodo colorato di rosso
G4VisAttributes* AnodoVisAttr = new
                 G4VisAttributes(G4Colour(1.0,0.0,0.0));
logicAnodo->SetVisAttributes(AnodoVisAttr);

// --- Catodo
solidCatodo = new G4Tubs("tubsCatodo",
               CatodoInnerRadius,CatodoOuterRadius,
               CatodoHeight/2,
               CatodoStartAngle,CatodoSpanningAngle);

logicCatodo = new G4LogicalVolume(solidCatodo,
                                  CatodoMaterial,
                                  "logCatodo");

phyCatodo = new G4PVPlacement(G4Transform3D(rot,    // ruota risp. al cubo
                          G4ThreeVector(CyliPosX,
                                        CyliPosY,
                                        CyliPosZ)),  // centrato con il cil.
```

```
                                        logicCatodo, // vol. logico
                                        "Catodo",    // nome
                                        logicCube,   // vol. madre
                                        false,       // no oper. booleane
                                        0);          // copy number

// catodo colorato di giallo
G4VisAttributes* CatodoVisAttr = new G4VisAttributes(
                                     G4Colour(1.0,1.0,0.0));
logicCatodo->SetVisAttributes(CatodoVisAttr);

// --- Mantello
solidMantello = new G4Tubs("tubsMantello",
                 MantelloInnerRadius,MantelloOuterRadius,
                 MantelloHeight/2,
                 MantelloStartAngle,MantelloSpanningAngle);

logicMantello = new G4LogicalVolume(solidMantello,
                                    MantelloMaterial,
                                    "logMantello");

phyMantello = new G4PVPlacement(G4Transform3D(rot,
       G4ThreeVector(CyliPosX,CyliPosY,CyliPosZ)),
       logicMantello,
       "Mantello",
       logicCube,
       false,
       0);

// mantello colorato di blu
G4VisAttributes* MantelloVisAttr = new G4VisAttributes(
                                        G4Colour(0.0,0.0,1.0));
logicMantello->SetVisAttributes(MantelloVisAttr);
...
}
```

5.5.2 Confronto fra simulazione e dati sperimentali

Completati i programmi di simulazione delle due configurazioni della camera Farmer, sono stati eseguiti i run necessari per il calcolo dell'output factor, per quantificare le differenze fra le due configurazioni geometriche.
Le condizioni della simulazione sono:

- $dept$ = profondita' pari a 83.35 mm di plexiglass ;
- $C = (5x5)\ cm^2$ con $N_5 = 6.25 \cdot 10^6$;
- $C = (10x10)\ cm^2$ con $N_{10} = 2.5 \cdot 10^7$;
- $C = (15x15)\ cm^2$ con $N_{15} = 5.625 \cdot 10^7$;

I valori di energia depositata ricavati con i due tipi di simulazione, semplificata e completa, sono riportati nella tabella seguente per tre campi.

En. Dep.	Farmer sempl	Farmer compl
Campo (5x5)cm2	0,720	0,570
Campo (10x10)cm2	0,725	0,729
Campo (15x15)cm2	0,770	0,691

Figura 5.21: Valori di energia depositata nella camera Farmer per tre diversi campi di radiazione, espressa in MeV.

Confrontando i valori ottenuti dalla simulazione con quelli misurati si possono derivare i coefficienti necessari per correggere l'approssimazione dovuta al numero di eventi, che non scala esattamente con l'area dei campi.

costanti K	Farmer sper	Farmer sempl	Farmer compl
Campo (5x5)cm2	0,896	1,244	1,571
Campo (10x10)cm2	1	1,379	1,371
Campo (15x15)cm2	1,058	1,372	1,532

Figura 5.22: Vengono riportati i coefficienti ricavati dal confronto fra i dati ottenuti dalla simulazione e quelli sperimentali.

Nella tabella (5.22) si riportano per i tre campi simulati i valori sperimentali e i coefficienti correttivi per le due simulazioni. Si osserva che i coefficienti correttivi sono dell'ordine del dieci percento. Inoltre i risultati relativi alla camera semplificata sono molto diversi rispetto a quelli ottenuti con la camera completamente simulata. Questo effetto suggerisce che il materiale che costituisce le pareti della camera gioca un ruolo importante nella formazione dell'output factor.

5.5.3 Confronto fra risultati di simulazione con lastre di diverso materiale

Un primo studio e' stato indirizzato per comprendere l'effetto del materiale della lastra forata che costituiva il gap. In questa configurazione il gap di aria e' sostituito da una lastra forata in cui i fori sono in corrispondenza dei pixel.

E' stato discusso nel precedente capitolo l'effetto della modifica in questione.
I profili, che nel caso del gap di aria tendevano ad appiattirsi con contributi importanti delle code, passando al gap con lastra di materiale con densita' uguale o superiore all'acqua erano in accordo con i risultati ottenuti con la Farmer. Allo stesso tempo, come gia' citato, l'accordo per quanto riguarda la misura dell'output factor migliorava con l'introduzione della lastra. Nondimeno la discrepanza fra Farmer e camera a pixel con lastra di vetronite rimane di qualche percento.

Nel presente paragrafo si discutono i risultati della simulazione con due materiali diversi della lastra: vetronite e plexiglass.

Ogni altra condizione e' stata mantenuta uguale e riportata qui di seguito.

- $\sigma = 5\ mm$ per il profilo del fascio;
- $N_5 = 6.25 \cdot 10^5$ per il campo $(5x5)\ cm^2$;
- $dept = 9\ mm$ la profondità in plexiglass;
- $Gap = 3\ mm$ di spessore;
- $Fori = 6\ mm$ di diametro.

Come si puo' rilevare la profondita' non corrisponde a quella normalmente considerata per la misura dello OF, ma a scopo di paragone i risultati sono ugualmente rilevanti. I valori di energia riportati nella tabella (5.23) sono la media dei sedici pixel centrali della camera. Nella stessa tabella vengono altresi' riportati i valori dello OF.

En. Dep.	Vetronite	Plexiglass	Vetronite	Plexiglass
Campo (5x5)cm2	0,054	0,057	0,853	0,729
Campo (10x10)cm2	0,056	0,069	1	1

Figura 5.23: Nelle prime due colonne sono riportati i valori di energia media depositata nei 16 pixel centrali della camera, espressa in MeV. Nelle ultime due gli stessi valori sono normalizzati rispetto al campo $(10x10)\ cm^2$.

I valori dello OF sono stati ottenuti normalizzando al campo $(10x10)\ cm^2$ per poter confrontare fra loro l'andamento delle curve ottenute dalle simulazioni, come mostra il grafico (5.24).
Confrontando plexiglass e vetronite si deduce che c'e' una differenza nella risposta mediamente di qualche percento. In particolare l'incremento dell'energia raccolta per campi di maggiore dimensione e' minore con la vetronite rispetto al plexiglass.

Sempre con la medesima configurazione si e' confrontato la risposta dei 16 pixel centrali della camera con vetronite ovvero con plexiglass in funzione della profondita'. I risultati sono riportati in Tabella 5.25 e in Figura (5.24). Si nota che le differenze fra le due configurazioni sono al piu' del 3% che corrisponde circa all'errore statistico della simulazione.

Possiamo sintetizzare le conclusioni come segue: lo sviluppo del deposito di energia in profondita' non dipende dal materiale della lastra, mentre invece lo OF pare cambiare sostanzialmente.

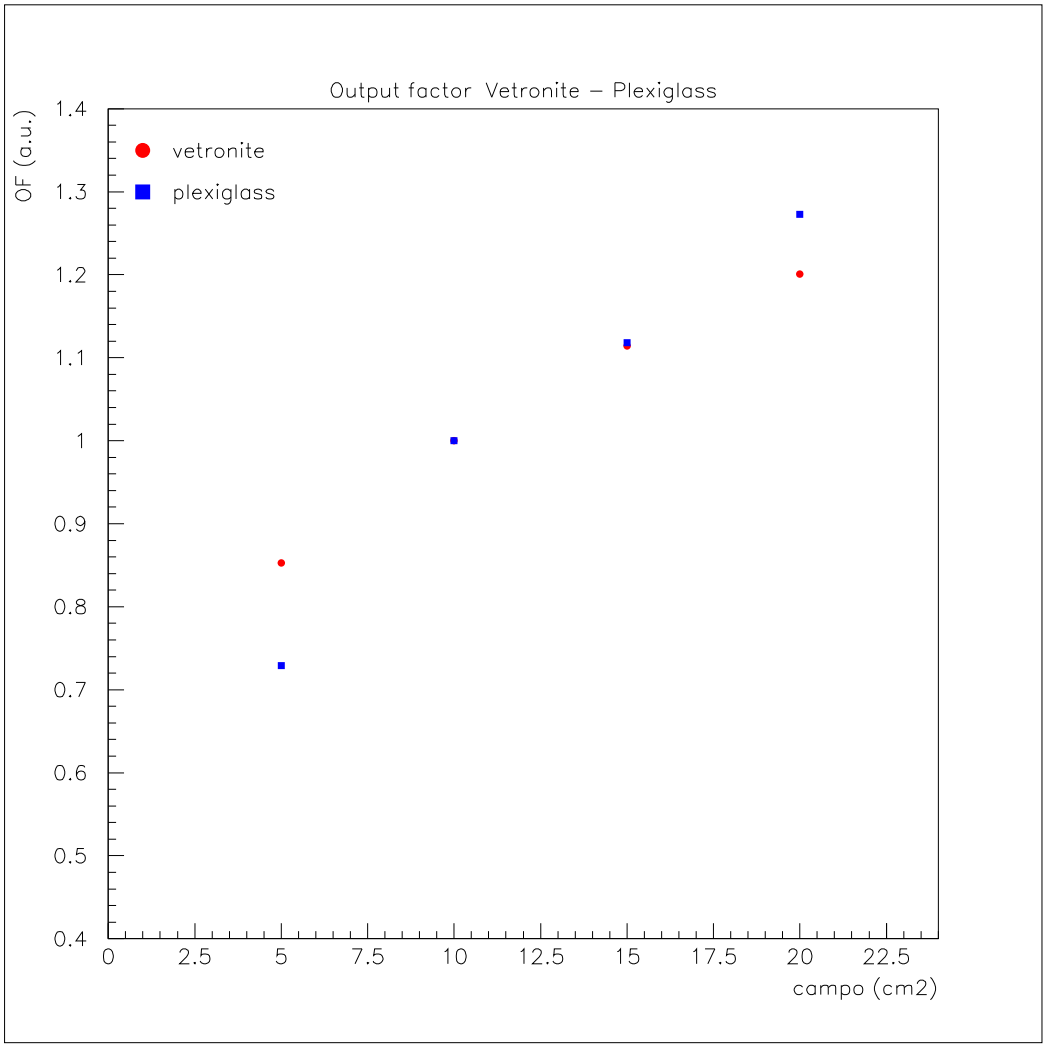


Figura 5.24: Confronto delle curve di output factor ricavate dalle simulazioni con lastra di vetronite e plexiglass rispettivamente. Lo OF è stato ottenuto per una profondità di 9 mm.

Dept (cm)	Vetronite	Plexiglass
5	0,522	0,512
10	0,421	0,420
15	0,329	0,337
20	0,273	0,255

Figura 5.25: Valori di energia depositati nella camera in funzione della profondita', espressi in MeV, per le due configurazioni rispettivamente con la griglia di vetronite e di plexiglass.

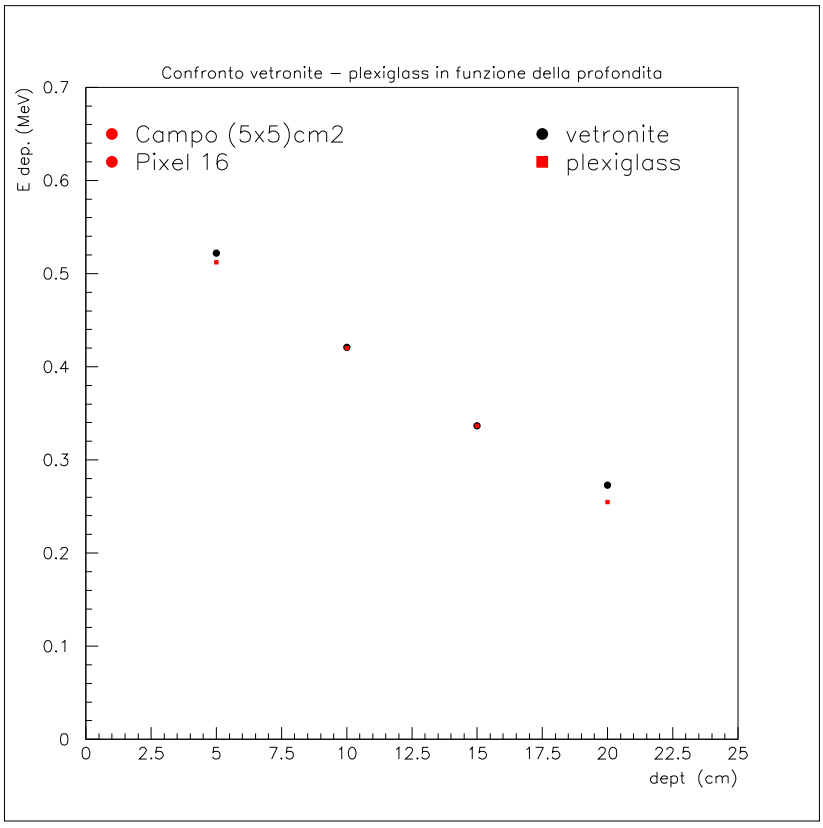


Figura 5.26: Confronto fra l'energia depositata nei 16 pixel centrali con la camera fornita di una griglia di vetronite (nero) o plexiglass (rosso).

5.5.4 Influenza della dimensione dei fori sul valore dello OF

Si e' proceduto simulando una lastra di plexiglass, di spessore pari a 5 mm, cambiando il diametro dei fori. Questo test permette di aumentare la quantità di materiale assorbitore fra un cameretta e l'altra e verificare così quali parametri influenzino lo OF. Sono stati effettuati dei run con la lastra di plexiglass con fori da 2 mm e 6 mm di diametro, ad una profondità di 83.35 mm in plexiglass, pari a 100 mm in acqua.

Le caratteristiche della simulazione sono:

- $\sigma = 5\ mm$ per il profilo del fascio;
- $N_5 = 6.25 \cdot 10^6$ per il campo $(5x5)\ cm^2$;
- $dept = 83.35\ mm$ la profondità in plexiglass;
- $Gap = 5\ mm$ di spessore;

I dati ricavati dalla media dei 16 pixel centrali sono riportati nella tabella (5.27) insieme con i valori normalizzati al campo $(10 \times 10)\ cm^2$.

En. Dep.	Fori 2 mm	Fori 6 mm	Fori 2 mm	Fori 6 mm
Campo (5x5)cm2	0,076	0,719	0,727	0,711
Campo (10x10)cm2	0,105	1,011	1	1

Figura 5.27: Energia media depositata nella camera per due diverse configurazioni, griglia con i fori da 6 mm e da 2 mm.

Si possono confrontare l'andamento delle curve di output factor della camera nelle due configurazioni descritte con quella della Farmer. Il confronto e' mostrato in figura 5.29, dal quale si rileva che il diametro dei fori non influenza il valore dello OF. Inoltre l'accordo con la simulazione della camera Farmer, effettuata per inciso nellse stesse condizioni, non e' soddisfacente. In particolare per il campo $(15x15)$ cm^2 il deposito di energia ottenuto e' largamente inferiore per la camera a pixel rispetto alla simulazione della Farmer.

OF (farmer compl)	Fori 2 mm	Fori 6 mm
Campo (5x5)cm2	0,833	0,815
Campo (10x10)cm2	1	1
Campo (15x15)cm2	0,835	0,838

Figura 5.28: Output factor calcolato attraverso le costanti di transizione della camera Farmer completa.

I valori dell'output factor presentati in figura 5.29 presentano un andamento apparentemente curioso, soprattutto per il campo $(15x15)$ cm^2 in cui l'energia depositata diminuisce notevolmente rispetto al campo $(10x10)$ cm^2. In effetti i valori suddetti per essere confrontati con i valori sperimentali debbono essere corretti ulteriormente per ugualizzare i flussi all'uscita della testata dell'acceleratore. La normalizzazione puo' essere fatta imponendo che i risultati della simulazione della Farmer coincidano con i dati sperimentali. In figura 5.28 si riportano i risultati corretti per la normalizzazione del flusso. Come si puo' rilevare la discrepanza del campo $(15x15)$ cm^2, pur rimanendo, e' diminuita.

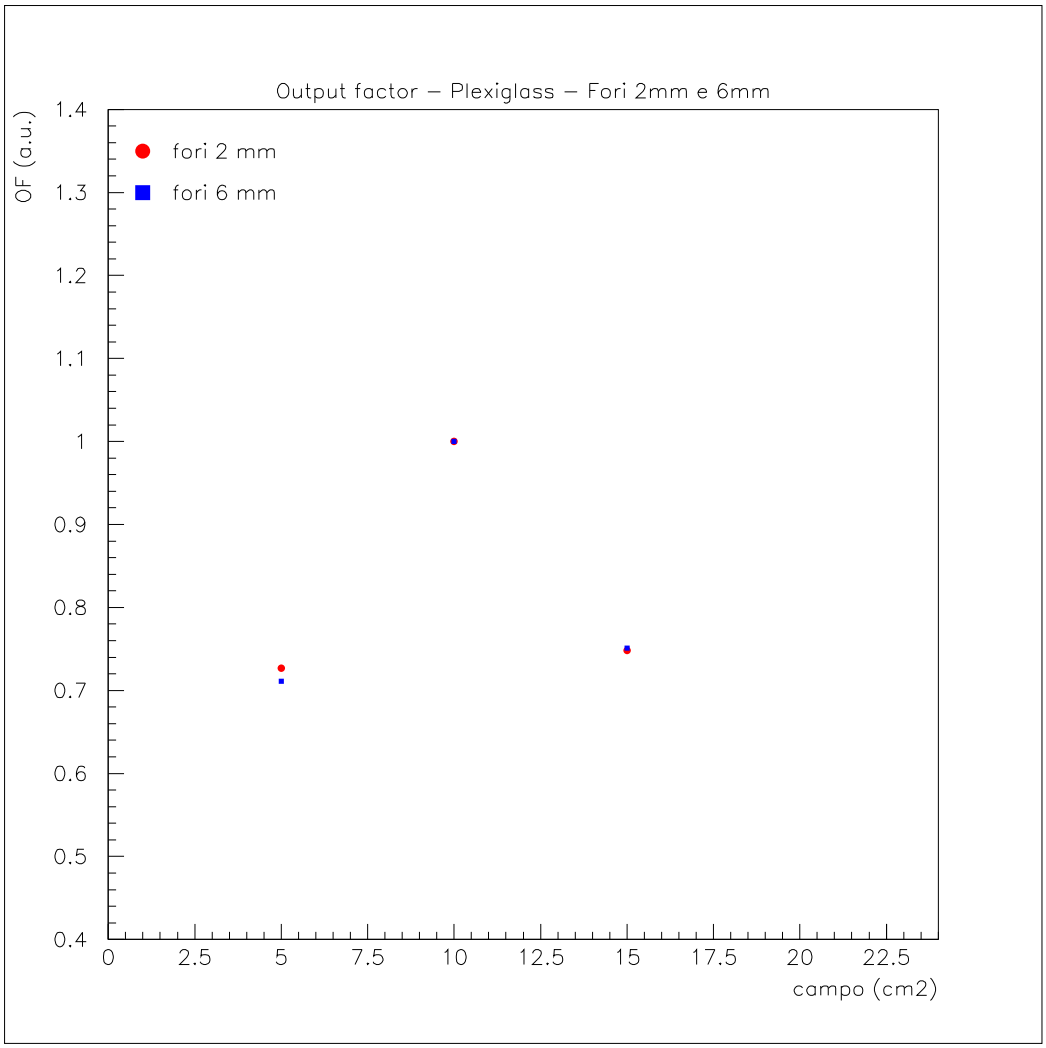


Figura 5.29: Andamento delle curve di output factor ricavate dalle simulazioni.

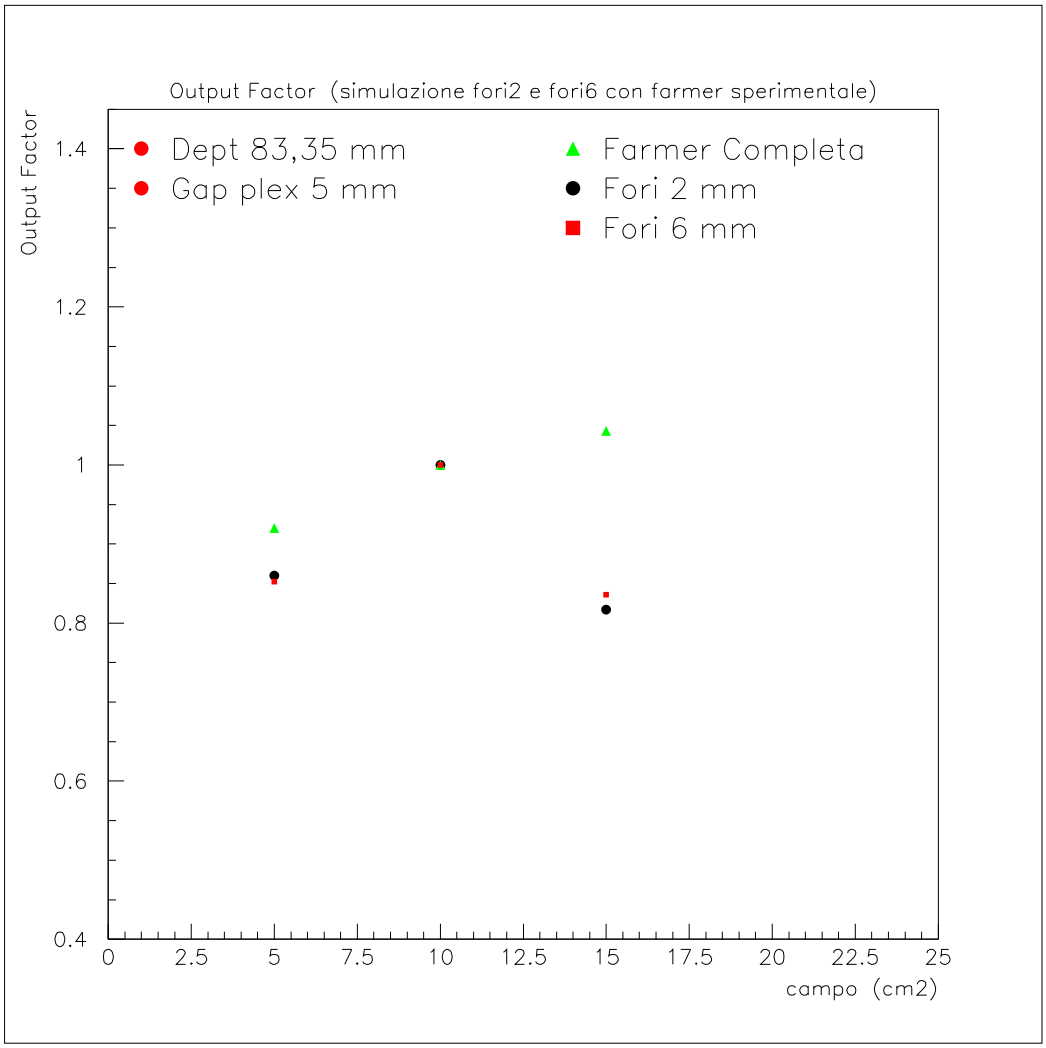


Figura 5.30: Andamento dei valori dell'output factor calcolati attraverso le costanti di transizione della camera Farmer completa.

5.5.5 Simulazione con un profilo trasverso del fascio a gradino

Il profilo trasverso del fascio, o in altri termini l'effetto dei collimatori, non e' noto dai costruttori dell'acceleratore. Inoltre una misura che fornisca spettro energetico e fluenza ai bordi del collimatore non e' di facile attuazione.
In questo paragrafo, si e' voluto investigare l'effetto di un diverso profilo del fascio ai bordi dei collimatori. Per lo piu' ci si aspetta che l'effetto, modificando la forma del fascio sul piano trasverso alla direzione dello stesso, sia maggiormente evidente nella misura dell'output factor.
L'effetto può essere associato ad una diversa distribuzione delle particelle incidenti all'interno del campo di radiazione. Ricordiamo che il numero N di fotoni inviati sul rivelatore era stato suddiviso in N_{flat}, numero di particelle contenute nel quadrato di lato $(L - \sigma)$ mm, ed N_{gaus}, numero di fotoni appartenenti alla penombra, contenuti nelle gaussiane laterali di semilarghezza σ [4].

Per studiare l'effetto si e' assunta la posizione estrema che e' quella di approssimare la forma del fascio ad un gradino. Si assume inoltre che il fascio abbia una distribuzione energetica uniforme su tutto il campo.
Si sono ripetute le simulazioni relative alla griglia in plexiglass con fori da 6 mm, a profondità 83.35 mm, per un numero di fotoni $N_5 = 10^6$. I dati ottenuti sono riportati nella tabella (5.31).

Fori 6 mm fascio quadro	En. Dep	En. Norm.
Campo (5x5)cm2	0,097	0,933
Campo (10x10)cm2	0,104	1
Campo (15x15)cm2	0,100	0,960

Figura 5.31: Valori di energia depositata e normalizzati ottenuti dalla simulazione della camera monitor a pixel, con nel gap la lastra forata di plexiglass e i fori di diametro pari a 6 mm, utilizzando un fascio incidente a sezione trasversale quadrata.

[4] Vedere capitolo 4, paragrafo 4.4.3.

Nelle identiche condizioni si è anche simulata la camera Farmer, per poter confrontare i dati ottenuti dalla simulazione della camera a pixel e per verificare quanto il profilo del fascio influisse sulle misure dell'output factor delle rispettive. In tabella sono riportati i valori di energia depositata e normalizzati al campo $(10x10)\ cm^2$.

Farmer fascio quadro	En. Dep	En. Norm.
Campo (5x5)cm2	0,743	1,107
Campo (10x10)cm2	0,671	1
Campo (15x15)cm2	0,702	1,046

Figura 5.32: Valori di energia depositata e normalizzati ottenuti dalla simulazione della camera Farmer completa irradiata con un fascio a sezione trasversa quadrata.

Confrontando i valori ottenuti dalla simulazione per l'output factor della camera Farmer con quelli misurati sperimentalmente si sono ricavati i coefficienti necessari alla corretta normalizzazione.
Nella tabella (5.33) sono stati riportati nella prima colonna i valori sperimentali dello OF della camera Farmer. Dal rapporto fra questi valori e quelli simulati si sono ricavati i fattori correttivi con i quali si e' prodotta la seconda colonna della stessa tabella. Questi rappresentano i valori simulati dello OF per il caso plexiglass con fori da 6 mm. Infine, nella terza colonna sono stati riportati i valori sperimentali ottenuti con lastra di plexiglass e fori da 3 mm.

L'accordo fra previsione da MonteCarlo e misure non e' buono e sostanzialmente non migliora rispetto alla configurazione del fascio con una penombra di 5 mm.
Come commento generale, indipendentemente dalla forma del fascio e dal tipo di lastra inserita, si possono raccogliere i possibili termini che contribuiscono alla discrepanza osservata:

1. incertezza statistica sulle simulazioni con la Farmer. Infatti avevamo valutato che l'errore statistico sul pixel centrale di lato 7.5 mm e spessore di 3

OF (fascio quadro)	Farmer sper.	Plex corretto	Plex sper.
Campo (5x5)cm2	0,896	0,755	0,871
Campo (10x10)cm2	1	1	1
Campo (15x15)cm2	1,058	0,973	1,087

Figura 5.33: La tabella riporta nella prima colonna le costanti di normalizzazione, ottenute dal confronto della simulazione della camera Farmer con i dati sperimentali, e, nella seconda colonna, i valori di output factor corretti sulla fluenza ottenuti dalla simulazione della camera con la lastra forata in plexiglass e i fori di 6 mm di diametro.

mm fosse di 2%. Assumendo che l'errore sia puramente dovuto al numero di particelle che depositano energia nel volume sensibile si avrebbe un valore di circa 2.5 % per la Farmer. Usando 16 pixel centrali e con una simulazione basata su 10^6 eventi nel campo $(5x5)$ cm^2 e a scalare per campi piu' grandi, si avra' un errore statistico di circa 1.5 %. L'errore complessivo (1.5 e 2.5 sommati in quadratura) danno un errore statistico di 3 %.

2. gli effetti a energie molto basse non sono stati investigati pienamente. L'analisi della raccolta di energia in funzione della profondita' appare indicare valori troppo elevati a basse energie. Questo effetto sistematico tende a influenzare particolarmente l'output factor.

5.6 Conclusioni

Al termine di questo lavoro di tesi giungiamo ad alcune considerazioni conclusive riguardanti la camera monitor a pixel utilizzata e il programma di simulazione implementato.

Il primo passo è stato quello di valutare l'**errore statistico** che dovevamo associare ad ogni misura, ottenuta dalla simulazione, caratterizzata da un numero N di eventi. Per un valore di N pari a $6.25 \cdot 10^6$, è stata calcolata una deviazione standard $\sigma = 0.9\%$, quando si media sui 4 pixel centrali della camera, $\sigma = 0.5\%$ quando si media sui 16 centrali. Questo è stato considerato l'errore associato ad ogni misura ottenuta con la simulazione. Qualora si volesse considerare il solo pixel centrale della camera l'errore sarebbe di circa il 2%. Si considera invece un errore di circa il 3% per le misure ottenute dalla simulazione della camera Farmer.

In seguito si è verificato l'**accordo fra i dati** della simulazione e quelli sperimentali, per assicurarsi che il programma implementato riproducesse in modo adeguato le condizioni reali delle misure effettuate con la camera. I profili sono in accordo entro il margine di errore dell'0.9%, anche se nei dati ricavati dalle simulazioni, aumentando il numero di eventi considerati almeno di un fattore 10, si potrebbe ottenere una maggior omogeneità fra i valori di energia depositata nei pixel centrali interessati dall'irradiazione.

E' stato successivamente studiato quanto l'implementazione dell'**allargamento gaussiano** del fascio migliorasse l'accordo con i dati sperimentali e si è concluso che le differenze erano trascurabili, perchè rientranti nel margine di errore delle misure. Dal confronto fra i valori ottenuti dalla simulazione della camera utilizzando un fascio con una penombra di 5 mm e uno con una penombra di 2.5 mm, possiamo affermare che l'accordo nelle code dei profili è leggermente migliore per il primo valore. Tutte le simulazioni sono state dunque eseguite con un allargamento gaussiano pari a 5 mm.

I dati ottenuti dalle simulazioni eseguite per il calcolo del **build-up** non sono in buon accordo con quelli sperimentali. Le simulazioni sono state eseguite per un numero limitato di spessori di plexiglass posti davanti alla camera, al fine di definire l'accordo con altre camere a ionizzazione. Per ottenere l'intera curva

dell'energia depositata in funzione della profondità si dovrebbero eseguire le simulazioni aggiungendo solo davanti alla camera 1 mm di plexiglass alla volta.
Dai dati ottenuti si può già concludere però che il codice Geant4 per energie molto basse, dell'odine di quelle da noi utilizzate, sovrastima la dose assorbita. I dati sperimentali della camera a pixel e della Farmer sono infatti in accordo fra loro, mentre la curva di dose in profondità ottenuta dalla simulazione è più alta circa del 25% ad una profondità di 150 mm.
Alcune simulazioni sono state mirate al calcolo dei profili di energia per un'irradiazione con un **gantry rotante** inclinato a 10^0, 20^0 e 30^0 gradi. I profili risultano in accordo con quelli sperimentali. Il confronto fra i profili a diverse angolazioni, ottenuti con la simulazione, mostra differenze più evidenti rispetto al confronto fra i dati sperimentali. Questo può essere causato o dal fatto che nella simulazione non si tiene conto dell'elettronica di lettura, che può uniformare il segnale in uscita dalla camera, o dal fatto che il simulatore tiene in conto anche il contributo di particelle a bassissima energia, misurando più carica di quella che effettivamente viene raccolta. Questo punto necessità di maggiori approfondimenti.

L'ultimo studio di questo lavoro di tesi è stato finalizzato alla misura dei coefficienti di **output factor**. E' stata presa come riferimento la camera Farmer, utilizzata in radioterapia per le misure di dosi. Sono state eseguite dunque delle simulazioni con la camera monitor a pixel in diverse configurazioni, per trovare quale di queste ci permettesse di ottenere dei coefficienti di output factor il più vicini possibile a quelli della Farmer. Il primo confronto è stato mirato alla scelta del materiale della griglia forata, presente fra l'anodo e il catodo della camera. I materiali presi in considerazione sono stati la vetronite e il plexiglass. Si è notato dal confronto che l'incremento dell'energia raccolta per campi di maggiore dimensione e' minore con la vetronite rispetto al plexiglass. E' stata poi inserita nella camera una griglia di plexiglass di spessore 5 mm, sono stati eseguiti dei run finalizzati a cercare il valore ottimale del diametro dei fori. Le simulazioni effettuate hanno mostrato che il diametro dei fori non influenza la misura dell'output factor. Servono dunque studi più approfonditi per determinare quali siano i parametri che intervengono nella dipendenza dell'energia depositata con il campo, e i valori ottimali che questi devono assumere per ottenere con la camera monitor a pixel dei dati in accordo con quelli misurati dalla camera Farmer.

Conclusione

Lo scopo di questo lavoro di tesi è stato il perfezionamento di un codice di simulazione di una camera monitor a pixel, progettata e realizzata dal gruppo TERA dell'INFN di Torino, e la verifica dei risultati ottenuti attraverso il confronto con i dati sperimentali.
Possiamo concludere dunque, da quanto esposto nei capitoli precedenti, che le misurazioni, finalizzate al calcolo dei profili dei fasci terapeutici e dell'energia da essi depositata in un dato volume sensibile, eseguite con la camera danno risultati soddisfacenti, in accordo con le altre acquisizioni ottenute da diversi tipi di camere a ionizzazione.
Il principale confronto è stato fatto con la camera Farmer, utilizzata in genere in ambito ospedaliero per le misure di dose e la calibrazione degli acceleratori, di cui è stato anche implementato un codice di simulazione per poter confrontare direttamente fra loro i dati ottenuti dal metodo MonteCarlo. La simulazione è stata creata con lo scopo di migliorare i parametri di costruzione, quali le dimensioni e i materiali, degli elementi principali che costituiscono la camera. E' necessario dunque che il codice riproduca fedelmente le condizioni reali dell'acquisizione dei dati, inizialmente in un assetto sperimentale semplice, poi via via più complesso e articolato per simulare anche situazioni di misura particolari.

La simulazione si è dunque arricchita nel corso dello studio attraverso delle fasi di sviluppo.
Il primo passo è stata la verifica che il codice implementato riproducesse fedelmente le condizioni reali, eseguendo il programma in un assetto sperimentale semplice e confrontando i profili dell'energia depositata nella camera con i dati sperimentali, acquisiti all'Ospedale San Giovanni Antica Sede di Torino, utilizzando un acceleratore lineare per fasci terapeutici di fotoni a 6 MeV.
Si è poi analizzata l'implementazione del fascio e delle sue caratteristiche fisiche per verificare quanto il suo profilo influenzasse il deposito di energia nella camera. I fasci terapeutici presentano infatti una minor intensità ai bordi rispetto al centro, fenomento dovuto al passaggio delle particelle nel collimatore, definito come *penombra*. Il codice è dunque stato implementato in modo da generare

un numero N_{flat} di particelle nella sezione trasversale quadrata del fascio, e un numero minore N_{gauss} sui bordi, secondo una distribuzione gaussiana. Sono state eseguite delle simulazioni per differenti valori della larghezza della gaussiana. I profili ottenuti presentano tra loro delle differenze trascurabili, inferiori all'errore statistico, pari allo 0.9%, che associamo ai dati ottenuti dalla simulazione.
Successivamente si è studiata la deposizione di energia in funzione della profondità per riprodurre il build-up caratteristico dei fotoni. I dati ricavati dalla simulazione non sono in accordo con quelli sperimentali della camera monitor a pixel, nè con quelli della camera Farmer. Possiamo ipotizzare che, essendo Geant4 un codice implementato per le simulazioni ad alta energia, quando si tratta con particelle poco energetiche la simulazione potrebbe non essere molto precisa. In particolare viene soprastimata la dose assorbita a differenti profondità.
Infine si sono eseguite delle misurazioni della dose assorbita in funzione del campo di radiazione, calcolando i coefficienti di output factor, definiti come il rapporto fra l'energia depositata nella camera per campi $(n\ x\ n)\ cm^2$ rispetto a quella relativa ad un campo di radiazione $(10\ x\ 10)\ cm^2$. A questo proposito è stata implementata la simulazione di un tipo di camera Farmer, modello IC10, per poter confrontare i dati ottenuti da essa con quelli relativi alla Farmer in diversi assetti sperimentali. Sono state studiate due differenti configurazioni di questa camera: una più semplice, costituita solo da un cubo di plexiglass in cui è contenuta una cameretta di aria, che definisce il volume sensibile; l'altra più completa, il quanto sono presenti tutti gli elementi necessari alla raccolta del segnale, quali l'anodo in alluminio, il catodo in grafite ed un sostegno, detto mantello, in plexiglass.
E' stata quindi studiata la differenza di energia depositata nelle due configurazioni e i rispettivi valori dell'output factor. Le curve hanno mostrato andamenti diversi, questo evidenzia il contributo della grafite presente nella camera.
Con i risultati della camera Farmer sono poi stati confrontati i dati delle simulazioni effettuate con la camera monitor a pixel in diverse configurazioni, è stato infatti eseguito il programma variando alcuni parametri della lastra forata, presente fra l'anodo ed il catodo della camera. Inizialmente è stato sostituito il materiale, da vetronite a plexiglass, ed aumentato lo spessore, in un secondo tempo il diametro dei fori della lastra.
Lo studio sulle curve dell'output factor può essere maggiormente approfondito cercando quali altri parametri influenzano in modo decisivo il deposito di energia nella camera in funzione del campo di radiazione.

Le simulazioni mirate a questi approfondimenti, devono utilizzare però un numero di eventi abbastanza grande da ridurre di molto l'errore statistico, perchè si devono investigare differenze dell'1%.

Concludendo i risultati ottenuti dalla simulazione e dall'acquisizione sperimentale con la camera monitor a pixel possono ritenersi soddisfacenti, in quanto in accordo con quelli misurati con il fantoccio ad acqua e con la camera Farmer, da sempre presa come riferimento per le misure di dose e considerata affidabile. Le uniche discrepanze riguardanti la simulazione si riferiscono alla misura della dose in funzione della profondità, sovrastimata dal codice di simulazione.

Per quanto riguarda la camera monitor a pixel invece l'unico disaccordo con i dati della camera Farmer si riferisce alla misura dell'output factor, per il quale è necessario continuare uno studio approfondito.
In futuro veranno analizzati altri parametri della camera, in particolar modo il materiale che la costituisce, poichè dai dati ricavati sembra che il disaccordo sull'output factor misurato non dipenda dalle componenti geometriche della camera quanto più dalla densità dei materiali che vengono in essa inseriti. Si deve infatti cercare un modo per fermare i fotoni di bassa energia prima del volume sensibile, poichè questi sono la causa della divergenza.

Infine si rileva che una simulazione completa ed abbastanza accurata necessita di tempi elaborazione estremamente lunghi. Risulta quindi laborioso poter simulare esattamente tutte le situazioni sperimentali, specie quando alcuni dei parametri sono cambiati di frequente.

Appendice A

Confronto fra il picco di Bragg nell'acqua e in una lastra di monomeri prodotto da un fascio monocromatico di protoni

Questo studio è stato effettuato in collaborazione con l'Istituto di Fisica di Roma, che svolge degli esperimenti per misurare la posizione del picco di Bragg di un fascio di protoni incidente su una lastra fotografica di monomeri polimerizzati. E' dunque stata implementata una simulazione del fantoccio ad acqua in cui viene inserita una lastra in posizione orizzontale, al fine di calcolare la profondità in cui viene rilasciata la massima energia. Sono dunque presentati in questa appendice una descrizione dell'esperimento realizzato, un commento al codice di simulazione che è stato implementato e i risultati ottenuti.

Introduzione

Per le misure di dose vengono utilizzati svariati tipi di dosimetri, basati su principi fisici differenti, sfruttati sempre però per la determinazione dell'energia depositata in un dato volume. L'utilizzo di lastre fotografiche fonda i suoi principi sulla reazione chimica che avviene nel materiale, cambiandone la densità ottica, quando questo è investito da un fascio di particelle, l'energia depositata dipende infatti dalla variazione del coefficiente ottico.

E' stato osservato sperimentalmente [Phis. Med. Biol. **44** (1999), pp. 1755-65] che la misura della dose dipendeva dall'orientamento delle lastre fotografiche. La misura veniva effettuata inserendo nel fantoccio ad acqua un pacchetto di 300 lastre, compresse in modo tale da poter supporre che fossero a contatto. Il pacchetto di lastre veniva posizionato parallelamente, oppure ortogonalmente rispetto al fascio. I risultati per fotoni da 6 MeV differivano del 15 %. In figura (34) vengono riportati i risultati presi dall'articolo di cui si e' detto prima in cui si

confronta la dose in funzione della profondita' per lastre parallele e perpendicolari per fotoni da 6 MV e per una sorgente di Co^{60}. Sono anche riportate le misure ottenute con una camera a ionizzazione. Dalla tabella si rileva che la lastra posta perpendicolarmente e' in buon accordo con i dati ottenuti con la camera a ionizzazione. Con la configurazione a lastre parallele la misura tende a soprastimare la dose.

Basandosi su questi principi, si sono sviluppati in seguito altri esperimenti, caratterizzati dall'utilizzo di differenti tipi di particelle, per diversi valori di energia, e da una impostazione sperimentale, riguardante le lastre fotografiche e la loro composizione chimica, che si discosta da quella originale.

Depth (cm)	Ion chamber	Perpendicular film	Parallel film
(*a*)			
0.5	100.0	100.0	100.0
2	93.0	93.0	93.0
5	78.3	77.5	86.1
10	55.9	56.5	68.9
15	39.0	37.6	51.7
20	27.0	26.6	39.0
25	18.6	18.8	30.2
(*b*)			
1.5	100.0	100.0	100.0
5	85.9	85.2	90.5
10	66.0	66.7	73.0
15	50.2	51.8	58.1
20	37.9	39.5	44.6
25	28.5	30.9	35.1

Figura 34: (a) Percentuale di dose assorbita per diverse profondità, misurata attraverso la dosimetria con lastre fotografiche e camere a ionizzazione, per un fascio di fotoni incidente generato da Cobalto^{60}. (b) Percentuale di dose assorbita per diverse profondità, misurata con lastre fotografiche e con una camera a ionizzazione, per un fascio di fotoni incidente di energia pari a 6 MeV.

.1 L'esperimento

L'esperimento condotto da un gruppo dell'Istituto di Fisica di Roma ha lo scopo di confrontare la posizione del picco di Bragg in una lastra orientata parallelamente ad un fascio di protoni, rispetto al picco di Bragg in una lastra posta perpendicolarmente al fascio. In entrambe le configurazioni le lastre sono immerse in un fantoccio ad acqua.
Nel caso in cui la lastra e' posta parallela al fascio, una frazione di protoni entrano dal bordo della lastra e la percorrono fino a fermarsi. Nel secondo caso la lastra si fa spostare in profondita' all'interno del fantoccio e ad ogni posizione si esegue una misura.
Il fascio utilizzato può essere considerato monocromatico, formato da protoni di 62 MeV, con una sezione trasversale quadrata di $(2.5x2.5)$ cm^2 e range medio[5] di circa 3.2 cm.
La lastra utilizzata è composta da due strati protettivi di plastica tra i quali vi è il gap sensibile costituito di monomeri (triphenylmethane leucodyes), che hanno la proprietà di polimerizzare se colpiti dalla radiazione. La densità della lastra è $d = 1.2\ g\ cm^{-3}$ e il numero atomico effettivo è 6.5.
La lastra è ottenuta dalla miscela di un elemento sensibile, che polimerizza, e di una base, necessaria per darle consistenza.
Le percentuali degli elementi presenti sono:

a) nella **soluzione sensibile**:

$H \Rightarrow 8.97\%$
$C \Rightarrow 60.58\%$
$N \Rightarrow 11.22\%$
$O \Rightarrow 19.23\%$

b) nella **base del film**:

$H \Rightarrow 4.196\%$
$C \Rightarrow 62.5\%$
$O \Rightarrow 33.302\%$

La lastra utilizzata è larga quanto il fantoccio ad acqua, spessa 278 μm e na-

[5] Con *range medio* si intende il cammino effettuato da N particelle di data energia in una sostanza, prima che la perdita di energia per ionizzazione non abbia ridotto il numero di particelle del 50%.

turalmente più lunga del range dei protoni in acqua.

Dall'esperimento è stato misurato un picco di Bragg per la configurazione del fantoccio con la lastra parallela al fascio circa del 30% più basso rispetto all'altro caso. La simulazione è stata implementata dunque per verificare se la differenza di energia depositata nelle due configurazioni dipendeva solo da fenomeni fisici relativi alle diverse densità dei due materiali o anche da reazioni chimiche che avvenivano durante la polimerizzazione. Ovviamente queste ultime non sono presenti nella simulazione.

.2 La simulazione

La simulazione dell'esperimento sopra descritto è stata implementata con una approssimazione: nel caso di lastra perpendicolare al fascio la lastra fotografica era sostituita da una lastra di acqua.

Lo scopo era di verificare se le differenze sperimentali fossero completamente descritte dalla simulazione che rende conto solo della diversa densità dei due materiali, oppure fosse necessario far intervenire delle reazioni chimiche che assorbivano parte dell'energia depositata. Attraverso una simulazione implementata con il toolkit di Geant4 infatti è possibile calcolare la quantità di energia depositata in funzione della profondità nelle due configurazioni sperimentali, tenendo in considerazione solo le interazioni fisiche che avvengono fra la materia e le particelle incidenti, e confrontarne i risultati.
Inoltre si vuole misurare il contributo dato dai protoni che rilasciano energia nell'acqua che circonda la lastra rispetto a quella depositata nella lastra stessa, per spiegare la presenza, nello spettro dell'energia in funzione della profondità, di due differenti picchi di Bragg di diversa altezza.
La simulazione implementata può essere divisa in due moduli: uno finalizzato alla descrizione del fantoccio ad acqua, di dimensioni $(30x30x45)\ mm^3$, suddiviso in $(30x30x150)voxel$, l'altro in cui viene definita una lastra di monomeri, centrata nell'origine e in posizione orizzontale, di $(30x0.278x45)\ mm^3$, divisa anch'essa in $(30x150)$ pixel.
La geometria e i materiali sono stati definiti nella classe *CameraDetectorConstruction* attraverso il seguente codice:

// definizione dei materiali

{

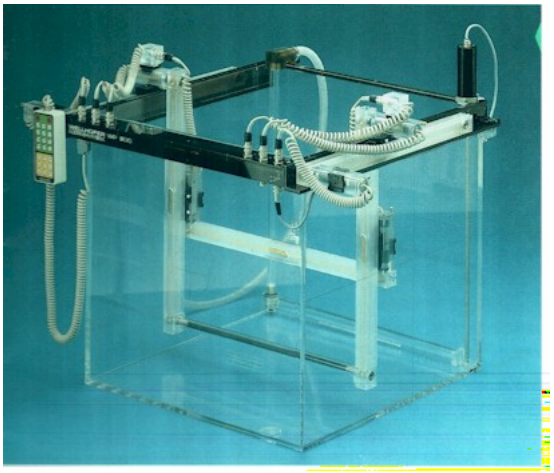


Figura 35: Foto di un fantoccio ad acqua. Si tratta di una vasca graduata, per poter effettuare misure in funzione della profondità, attrezzata in modo tale da permettere l'inserimento di dosimetri standard, come la camera Farmer.

```
...
// acqua
density = 1.000*g/cm3;
G4Material* H2O = new G4Material(name="Acqua",
 density,
 ncomponents=2);
H2O->AddElement(H, natoms=2);
H2O->AddElement(O, natoms=1);

// monomeri
density = 1.200*g/cm3;
G4Material* Mon = new G4Material(name="Monomeri",
  density,
  ncomponents=4);
Mon->AddElement(H, fractionmass = 0.0897);
Mon->AddElement(C, fractionmass = 0.6058);
Mon->AddElement(N, fractionmass = 0.1122);
Mon->AddElement(O, fractionmass = 0.1923);

// film sensibile
density = 1.200*g/cm3;
```

```
 G4Material* film = new G4Material(name="film",
   density,
   ncomponents=3);
 film->AddElement(H, fractionmass = 0.04196);
 film->AddElement(C, fractionmass = 0.625);
 film->AddElement(O, fractionmass = 0.33302);
 ...
}
```

// definizione del fantoccio

```
{
 ...

 // ciclo sui tre assi
 for(G4int i=0;i<NbOfPixelZ;i++){
    for(G4int j=0;j<NbOfPixel;j++) {
       for(G4int k=0;k<NbOfPixel;k++) {
  // posizionamento dei voxel
          G4int n=i*NbOfPixel*NbOfPixel+j*NbOfPixel+k;
  G4double PixelPosZ =(-((NbOfPixelZ/2-0.5)*(PixelDistZ))+i*PixelDistZ);
          G4double PixelPosY =(-((NbOfPixel/2)*(PixelDist))+j*PixelDist);
  G4double PixelPosX =(-((NbOfPixel/2-0.5)*(PixelDist))+k*PixelDist);

  // volume solido
  solidPixel[n] = new G4Box("boxPixel",
      PixelSizeXY/2 ,PixelSizeXY/2,PixelThickness/2);
  // volume logico
  logicPixel[n] = new G4LogicalVolume(solidPixel[n],
      PixelMaterial,
      "logPixel");

  // volume fisico
  phyPixel[n] = new G4PVPlacement(0,
    G4ThreeVector(PixelPosX,PixelPosY,PixelPosZ),
    logicPixel[n],
    "Pixel",
    logicWorld,
    false,
    n);
```

```
  // colorato in viola
  G4VisAttributes* PixelAttr= new G4VisAttributes(G4Colour(0.5,0.3,0.5));
  logicPixel[n]->SetVisAttributes(PixelAttr);

       } // for(k)
    } // for(j)
 } // for(i)
```

// definizione della lastra

```
 // ciclo sui due assi x e z
 for(i=0;i<(NbOfPixelZ);i++){
    for(G4int j=0;j<(NbOfPixel);j++){
// posizionamento dei pixel
     G4int n=i*NbOfPixel+j;
     G4double LastraPosY = 0.0;
     G4double LastraPosX = (-((NbOfPixel/2-0.5)*(PixelDist))+j*PixelDist);
     G4double LastraPosZ = (-((NbOfPixelZ/2-0.5)*(PixelDistZ))+i*PixelDistZ);

// volume solido
     solidLastra[n] = new G4Box("Lastra",
     LastraSizeX/2,LastraSizeY/2,LastraSizeZ/2);

// volume logico
     logicLastra[n] = new G4LogicalVolume(solidLastra[n],
       LastraMaterial,
     "Lastra");

// volume fisico
     phyLastra[n] = new G4PVPlacement(0,
   G4ThreeVector(LastraPosX,LastraPosY,LastraPosZ),
   logicLastra[n],
   "Lastra",
   logicWorld,
   false,
   n);

// colorato di azzurro
G4VisAttributes* LastraAttr= new G4VisAttributes(G4Colour(0.5,0.5,0.5));
```

```
logicLastra[n]->SetVisAttributes(LastraAttr);
    } // for(j)
 } // for(i)
 ...
}
```

Il simulatore restituisce all'utente tre file HBOOK contenenti le distribuzioni di energia depositata nei rispettivi voxel, in funzione di cosa vuol essere misurato.

Si otterrà dunque:

- Un istogramma lungo i $(30x30)$ pixel posti sul piano perpendicolare all'asse del fascio, che permette di studiare la sua distribuzione spaziale e le caratteristiche della sua sezione trasversale. Sull'asse z vengono riportati i valori dell'energia depositata lungo tutto il percorso della particella nel fantoccio, vengono cioè sommate, pixel per pixel, le dosi rilasciate in ogni strato di 300 μm di acqua.

- Un numero N, inserito come parametro dall'utente, di istogrammi, riportanti l'energia depositata nei $(30x30)$ pixel posti sul piano perpendicolare all'asse del fascio, relativi ai primi N strati di 300 μm di acqua. Questi permettono di verificare le differenze di energia depositata al centro e ai bordi del fascio a diverse profondità.

- Un istogramma di $(30x150)$ pixel, riportante l'energia depositata lungo l'asse z, dunque nella direzione del fascio. Questo istogramma è stato utilizzato per calcolare il picco di Bragg lasciato dai protoni sul piano che divide a metà il fantoccio.

.3 Risultati

La simulazione è stata eseguita modificando il campo di radiazione rispetto ai valori sperimentali, allo scopo di ottimizzare l'effetto delle particelle incidenti attorno al volume sensibile e quindi velocizzare l'elaborazione dei dati.
Il campo utilizzato nell'esperimento misura $(2.5x2.5)$ cm^2, ma, essendo la lastra solo spessa $300\mu m$, tutti i protoni distanti da essa oltre qualche mm vanno persi senza contribuire all'annerimento del gap sensibile. Per evitare dunque di inviare

particelle dove il deposito di energia non viene tenuto in considerazione[6] e quindi di allungare notevolmente il tempo della simulazione, si è implementato un fascio di dimensioni $(25x4)\ mm^2$. Così facendo si è supposto che i protoni influenzino il valore di dose assorbita nella lastra entro $\pm 2\ mm$, un'approssimazione che consideriamo accettabile.

Le condizioni della simulazione sono:

- $N = 10^5$, numero di protoni;
- $E_p = (62 \pm 0.1)\ MeV$, energia delle particelle;
- $P_0 = (0, 0, -50)\ mm$, origine del fascio;
- $C = (25x4)\ mm^2$, dimensioni del campo.

I plot, mostrati in figura (36), ottenuti dalle due simulazioni mostrano come risulti più 'alto e snello' il picco di Bragg relativo ai protoni in acqua, rispetto a quello più 'basso e panciuto' della lastra. Questo è in accordo con il fatto che la densità nella lastra sia maggiore e quindi il picco sia situato ad una profondita' minore.

Per misurare la posizione del picco di Bragg nelle due configurazioni, rispettivamente nell'acqua e nella lastra, sono state sommate le 20 file centrali di pixel[7] investite dal campo e visualizzate lungo la direzione del fascio. I profili così ottenuti sono mostrati nelle figure (37) e (38).
Da questi si misura la posizione del picco di Bragg nelle due configurazioni:

in acqua[8] $\Rightarrow 32\ mm$;
nella lastra $\Rightarrow 31\ mm$.

Il valore ottenuto nell'acqua è in accordo con la misura sperimentale.
Nel caso della lastra parallela al fascio si debbono considerare due contributi per spiegare la posizione del picco:

[6] Ricordiamo che nella simulazione con Geant4 si definisce un volume sensibile. Vengono seguite dal programma solo le particelle che lo attraversano e viene misurata solo l'energia depositata in esso.

[7] Si ricorda che la simulazione è stata fatta per una matrice di (30x30x150) pixel, sono quindi state tralasciate le prime e le ultime 5 file.

[8] Ricordiamo che la simulazione del fantoccio ad acqua simula la lastra fotografica perpendicolare al fascio

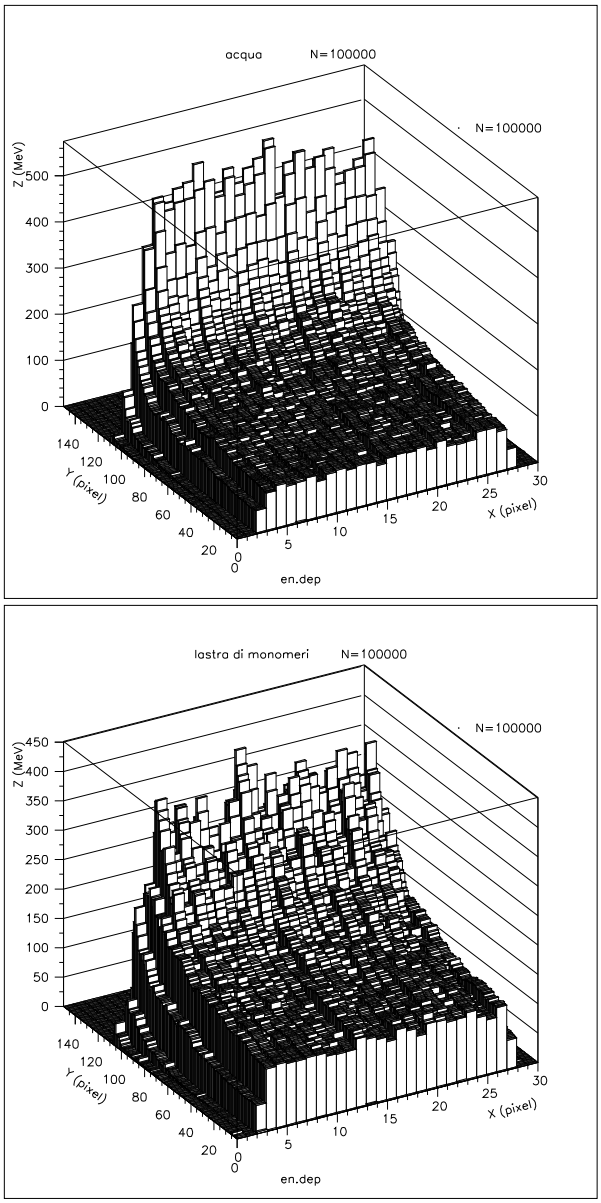


Figura 36: Istogramma 2D rappresentante l'energia depositata in una matrice di (30x150) pixels. I due plot si riferiscono alla simulazione nel fantoccio ad acqua e nel fantoccio con, in aggiunta, la lastra.

1. contributo dei protoni che rimangono nella lastra lungo tutto il percorso e che si fermano prima a causa della densita' maggiore della lastra rispetto all'acqua;

2. contributo dei protoni che viaggiano nell'acqua in modo radente alla lastra e che vi entrano per scattering multiplo lungo il loro percorso. Questi protoni hanno un range che dipende dalle condizioni di percorso e quindi allargano notevolmente il picco.

La somma dei due contributi si presenta come un picco ancora evidente dovuto ad 1) preceduto da un picco allargato dovuto a 2).
Questo è giustificato dal fatto che la densità è maggiore [9] e dunque i protoni sono maggiormente rallentati lungo il loro cammino.

Infine vengono confrontati in figura (39) i profili ottenuti dalla simulazione nelle due configurazioni, per confrontare fra loro i valori dei picchi, sommando le 4 file centrali di pixel.
Per la lastra otteniamo un valore di energia, espresso in unità arbitrarie, pari a 1500, rispetto a quello misurato nell'acqua, pari a 2125.

.4 Conclusione

Dai risultati della simulazione si può concludere che la differenza fra i picchi di Bragg ottenuti nell'acqua e in una lastra di monomeri polimerizzanti è di circa il 30 % simile al valore ottenuto sperimentalmente.
Si potrebbe quindi concludere che se avvengono delle reazioni chimiche durante l'irragiamento della lastra, queste hanno un effetto trascurabile sull'energia depositata.

[9] La densità dell'acqua è 1 $g\ cm^{-3}$, quella della lastra è stata misurata pari a 1.2 $g\ cm^{-3}$.

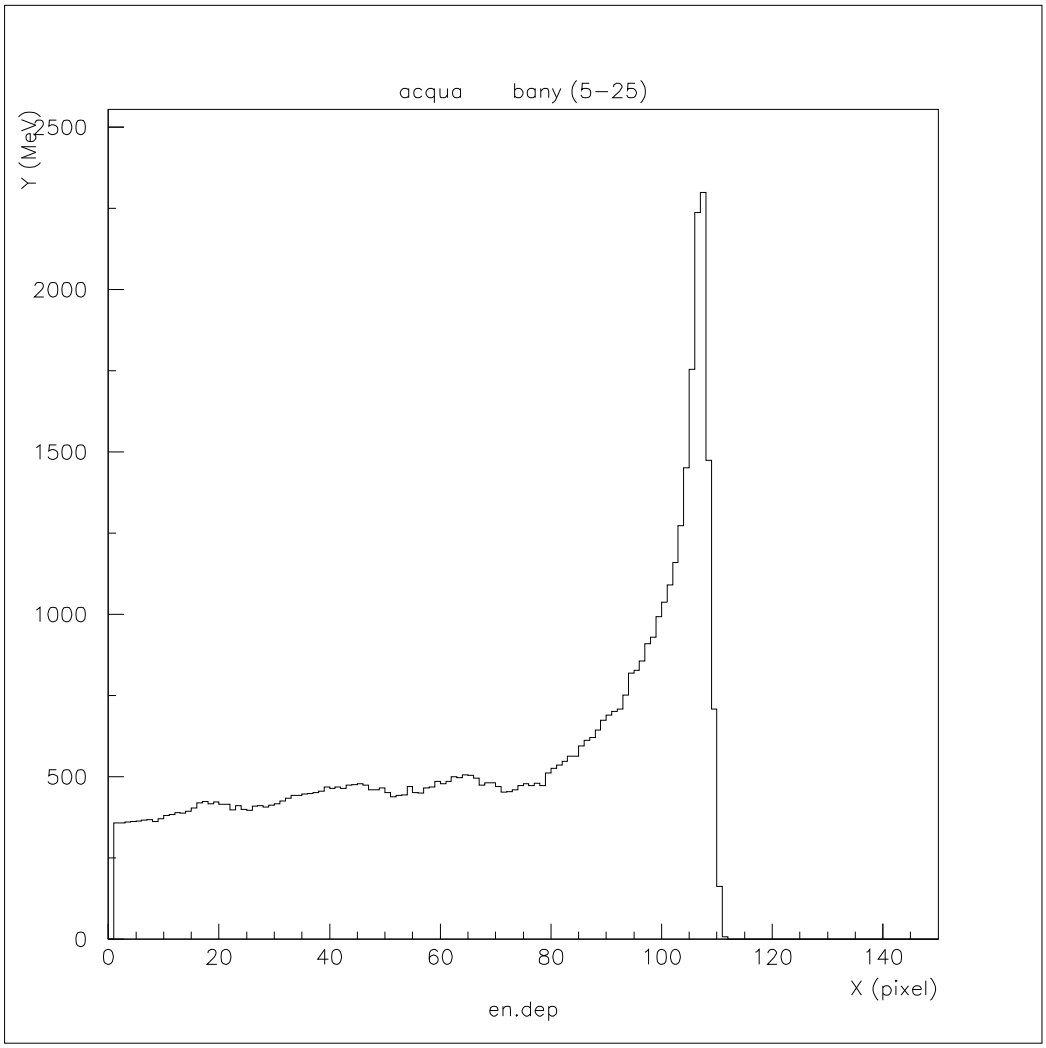


Figura 37: Profilo dell'energia depositata lungo la direzione del fascio dai protoni in acqua, ottenuto dalla somma delle 20 slice centrali.

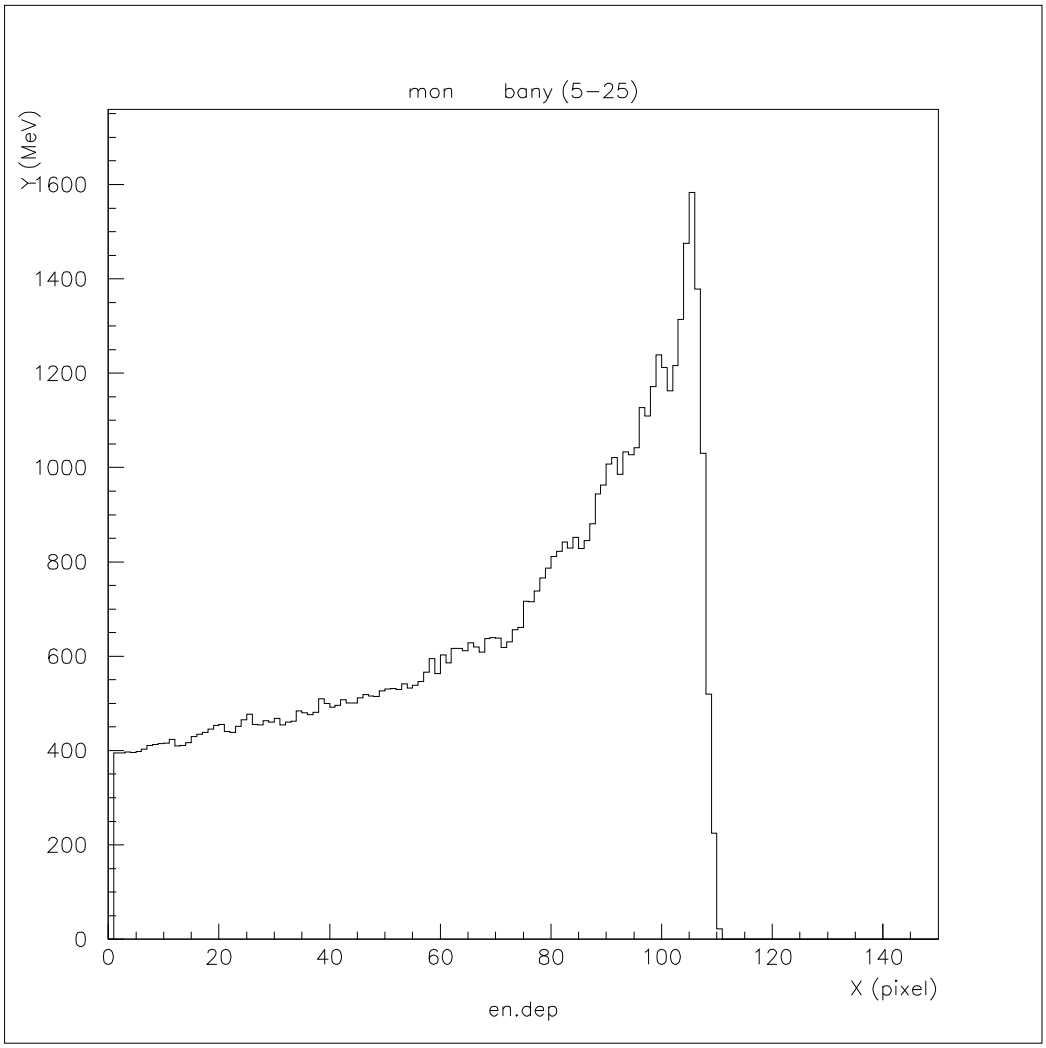


Figura 38: Profilo dell'energia depositata lungo la direzione del fascio dai protoni nella lastra di monomeri, ottenuto dalla somma delle 20 slice centrali.

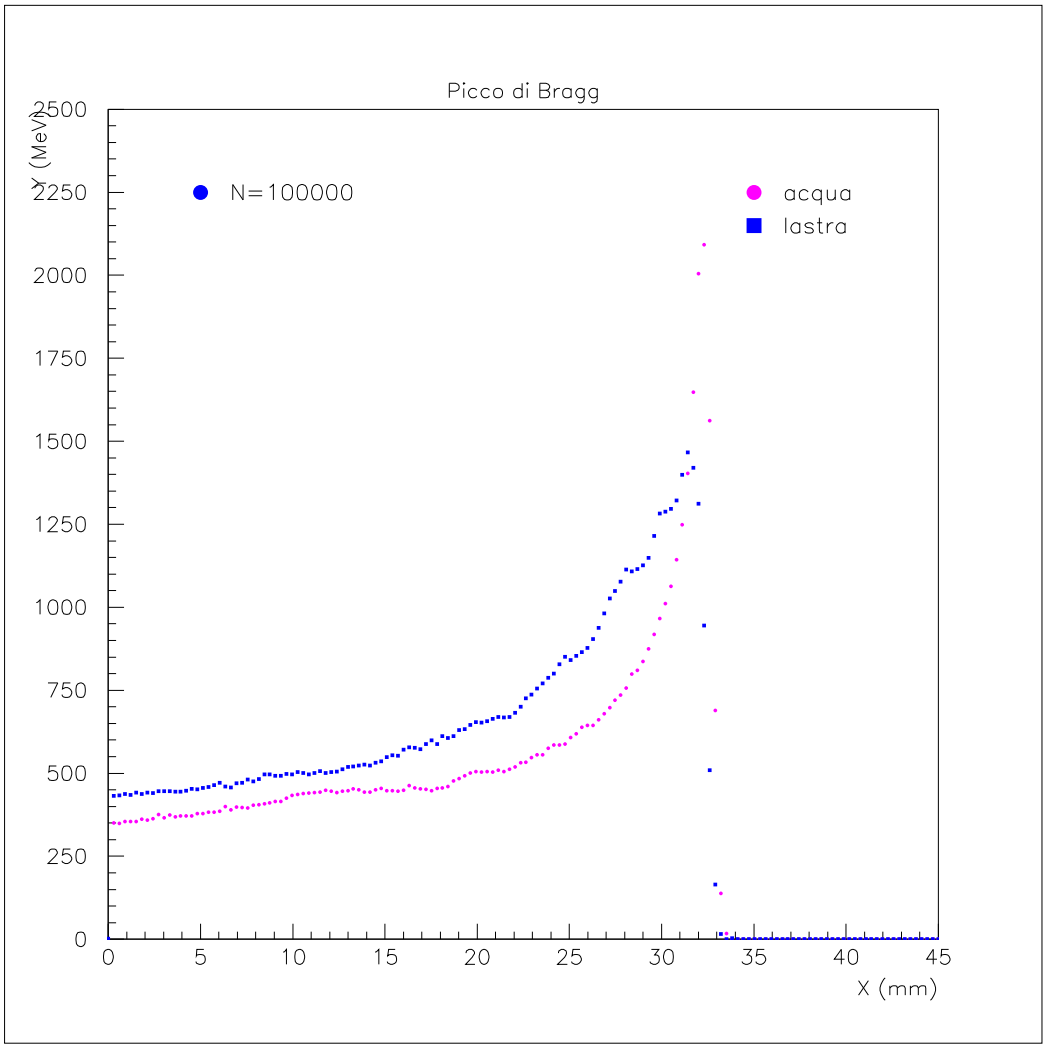


Figura 39: Confronto fra i profili della dose, in funzione della profondità, ottenuti nell'acqua e nella lastra, sommando le 4 file centrali.

Appendice B

Progettazione ed Implementazione di una interfaccia grafica per l'esecuzione di simulazioni

In questa appendice viene presentato un programma, implementato con il linguaggio JAVA, che permette di modificare in modo semplice e veloce, attraverso delle finestre di dialogo, i parametri che caratterizzano una simulazione ed eseguire dei run. Vengono commentate le classi costruite, i metodi utilizzati, viene descritta la modalità di utilizzo dell'interfaccia ed i possibili sviluppi futuri.

Introduzione

Ogni simulazione implementata con il codice Geant4 necessita di un file macro[10], definito per default nel *main()* come 'prerun.mac', nel quale vengono impostate delle righe di comando proprie di Geant che servono all'avvio del run. Lo scopo di questa interfaccia grafica è permettere ad un utente, non a conoscenza dei particolari dell'implementazione del simulatore, di poter eseguire dei run, impostando le variabili da definire a suo piacimento, e leggerne i risultati.
Inoltre ora la struttura dell'interfaccia e' semplice, in quanto sono state implementate le funzionalita' primarie, ma le possibilita' di svilupparla e completarla sono molte. Ad esempio si potrebbe implementare un modulo che permetta di visualizzare i risultati ottenuti dalle simulazioni e poterli quindi mostrare su qualsiasi macchina.

E' stato scelto il linguaggio di programmazione JAVA soprattutto per le sue caratteristiche di portabilità su piattaforme differenti, è cioè indipendente dal sistema operativo su cui opera. Questo significa che per poter eseguire lo stesso codice java su macchine diverse è solamente necessario disporre della macchina virtuale java *(JVM)* e quindi non occorre effettuare ulteriori compilazioni.

[10] Vedere a proposito il capitolo 3, ultimo paragrafo.

.1 Concetti di base

Come già accennato in precedenza, questa interfaccia è stata progettata al fine di fornire uno strumento che consenta una facile configurazione di un file macro che per quanto riguarda le simulazioni è solitamente costituito da un insieme di run.
In particolare il sistema mette a disposizione le seguenti funzionalità:

- aprire e visualizzare il contenuto di un file macro esistente;
- modificare i comandi del file e salvare la nuova macro creata;
- operare modifiche a livello di run [11] e quindi gestire una sequenza di simulazioni;
- eseguire la simulazione fornendo i dati macro inseriti;
- visualizzare i risultati prodotti da Geant4.

Durante la fase di progettazione dell'interfaccia grafica si è focalizzata l'attenzione sul fatto di rendere tutte le informazioni subito visibili all'utente in modo da facilitarne l'operabilità.
L'interfaccia è divisa in tre settori principali:

- un'area per la visualizzazione dell'elenco dei comandi del file macro che necessitano l'impostazione di uno o più parametri;
- un'area per la visualizzazione di tutte le configurazioni effettuate dall'utente separate per singoli run;
- un'area per la visualizzazione dei risultati prodotti da Geant4.

Il sistema è stato strutturato utilizzando la metodologia Object Oriented e nella figura (40) viene mostrato il diagramma UML delle classi di cui è composto. Dalla classe principale *Simulation* dipendono le altre classi implementate, utilizzate per l'apertura di finestre di dialogo, come *DialogCommand* e *DialogHelp*, per la lettura di dati, come *ReadData*, o per la creazione e memorizzazione di comandi (classe *Command*).

[11] Si noti che per run si intende un insieme di comandi che fornisce l'input per una singola esecuzione della simulazione con Geant4.

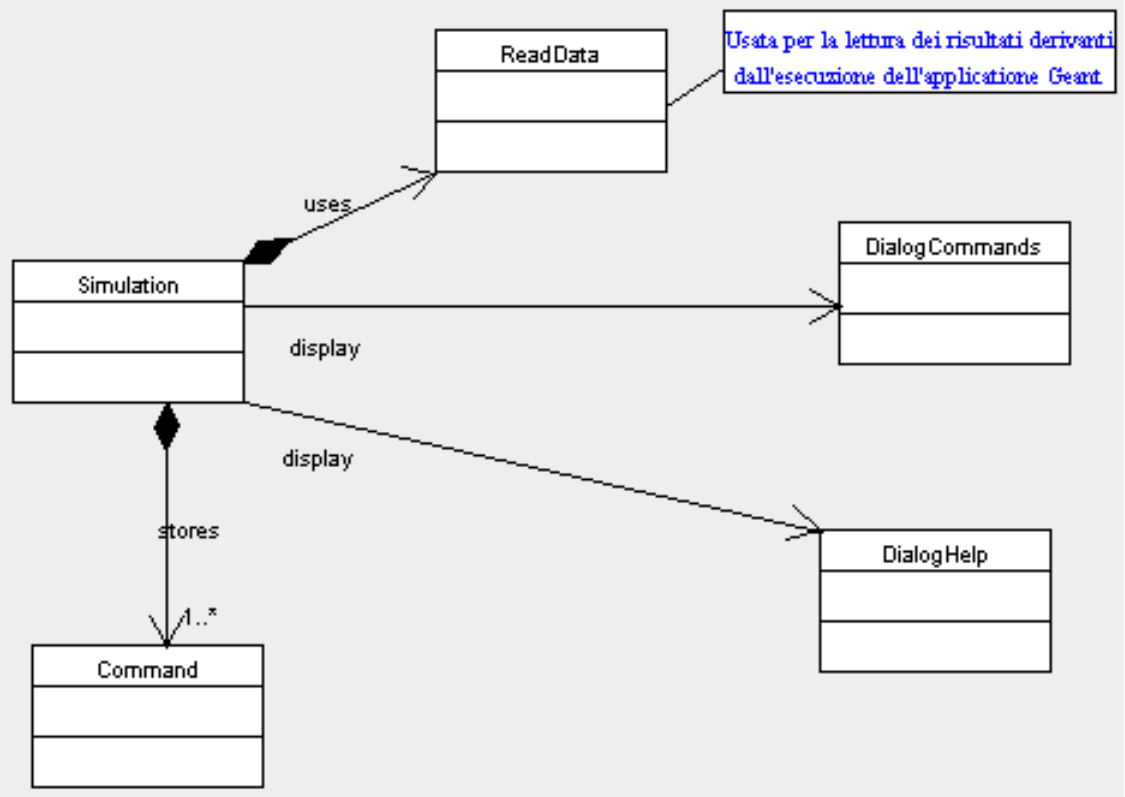


Figura 40: Diagramma UML delle classi implementate per lo sviluppo dell'interfaccia grafica.

.2 Implementazione

Il programma è centrato sulla classe principale *Simulation*, estensione della classe Java *JFrame* del package javax.swing. Tale classe costruisce la finestra di base dell'interfaccia e inserisce i principali componenti. In essa è contenuto il *main()* in cui viene creata un'istanza di Simulation a cui viene applicato il metodo *show()* per la visualizzazione.

```
public static void main(String[] arg)
{
    // crea l'oggetto Simulation
    Simulation simulation  = new Simulation();
    simulation.show();
} // main()
```

Nella classe *Simulation* sono stati implementati i seguenti metodi principali:

- **private File openFile()**, che restituisce un file aperto tramite una finestra di dialogo;
- **private Vector readMacFile(File f)**, che legge un file macro e restituisce un vettore i cui elementi rappresentano singoli comandi;
- **private boolean parseCommands(Vector v)**, che esegue il parsing di ogni comando e restituisce *true* se tutti i comandi sono stati riconosciuti;
- **private boolean saveFile(File f)**, che permette di salvare il nuovo file macro creato;
- **private void addRun(String runName)**, che aggiunge un nuovo run alla sequenza di runs construita;
- **private void removeRun()**, elimina il run corrente, selezionato nella lista dei runs.

La classe *Command* rappresenta un singolo comando del file macro. Questo è caratterizzato da un nome e da una sequenza di parametri che definiscono il materiale, i valori e le rispettive unità di misura. In questa classe sono implementati i metodi per l'impostazione e la lettura dei valori delle variabili d'istanza.

La classe *DialogCommands*, estensione della classe *javax.swing.JDialog*, consente la modifica dei parametri di ogni singolo comando attraverso una finestra di dialogo. L'elenco di tutti i materiali a disposizione viene caricato da un file testuale

e quindi facilmente configurabile a seconda delle necessità.

La classe *ReadData*, estensione della classe *java.lang.Thread*, è stata implementata per leggere in modo asincrono l'output della simulazione e lo stampa in nell'area di testo dell'interfaccia *Console Level*.

.3 Utilizzo dell'interfaccia

Verrà presentato ora un esempio di utilizzo dell'interfaccia grafica per creare un nuovo file macro ed eseguire la simulazione di Geant4. Per avviare l'interfaccia grafica è necessario eseguire da console il comando *java Simul.Simulation*, dove Simul è il package in cui sono inseriti i file .class dell'applicazione. La prima videata è mostrata in figura (41). Si può notare la toolbar e le tre aree da cui è composta, in precedenza descritte.

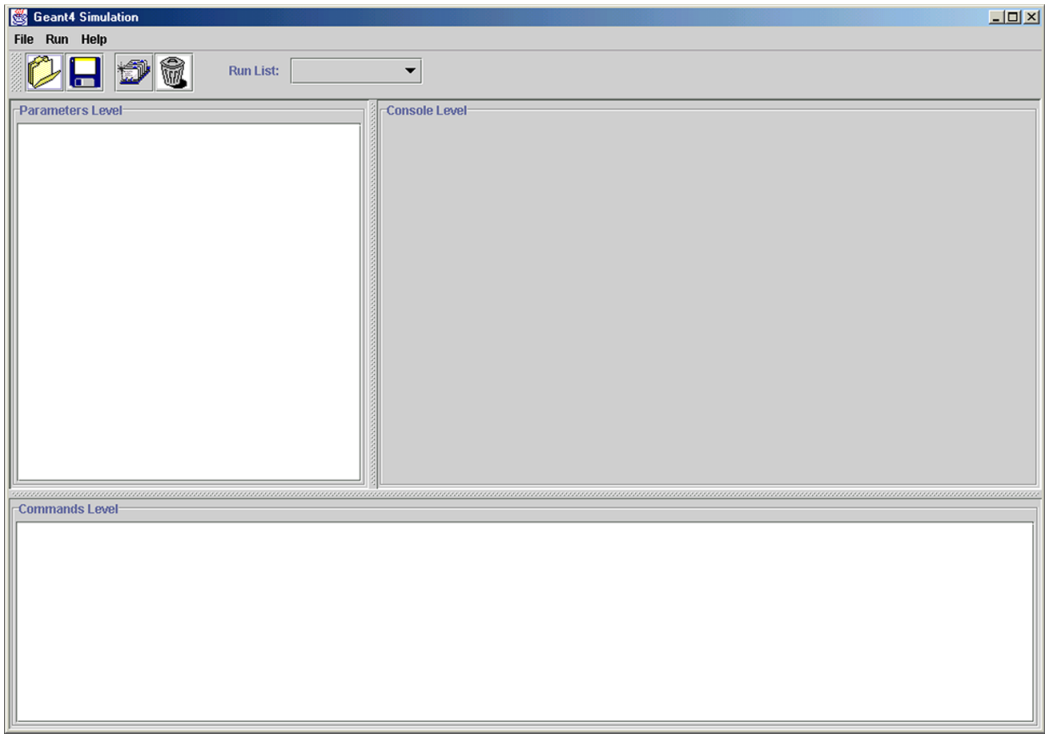


Figura 41: Videata iniziale dell'interfaccia.

Selezionando l'apposito bottone sulla toolbar o attraverso il menu 'File > Apri', viene aperta la finetra di dialog di sistema che permette di ricercare e selezionare il file macro desiderato. Effettuata la scelta, il file viene letto ed i singoli comandi interpretati e visualizzati come riportato nella figura (42).

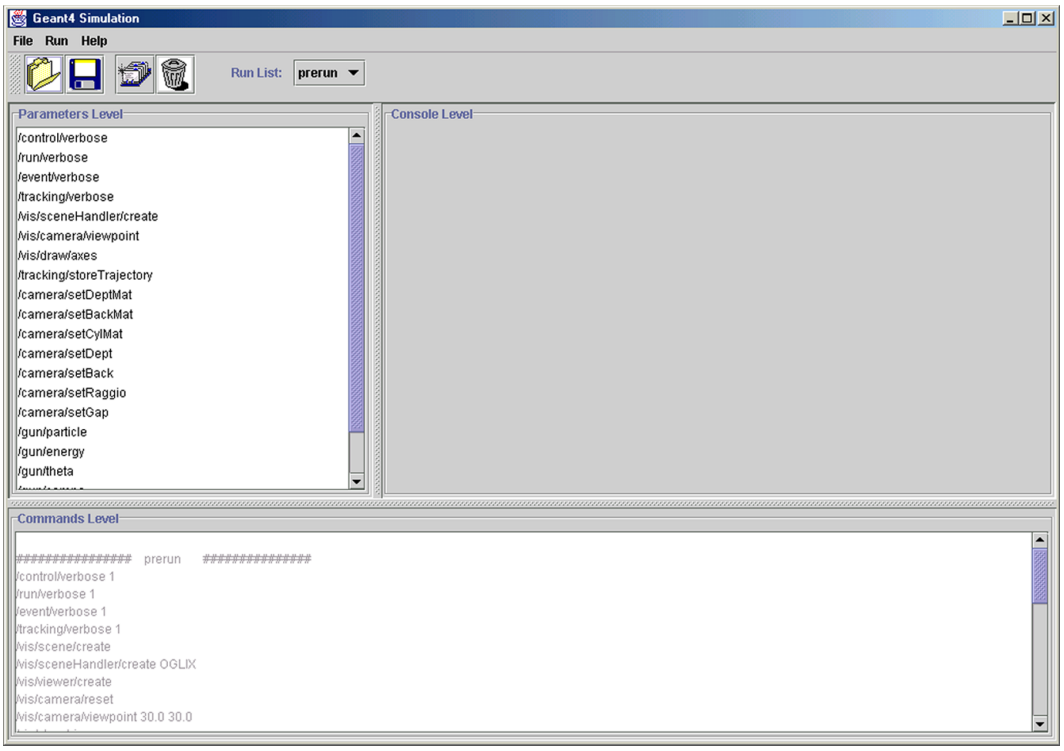


Figura 42: Visualizzazione comandi file macro.

La configurazione dei parametri di un dato comando viene effettuata selezionandolo dall'area *Parameters Level*, attraverso la finestra di dialogo (43). Viene in seguito presentata la sequenza di operazioni necessarie per l'impostazione del materiale, dei valori e dell'unità di misura.
Dopo l'aggiunta di un nuovo run, effettuata attraverso il bottone in toolbar o il menu 'Run > Add', viene presentata la seguente finestra (46), in cui si ha la possibilità di selezionare e modificare il run desiderato, selezionandolo attraverso il combo box presente nella toolbar.

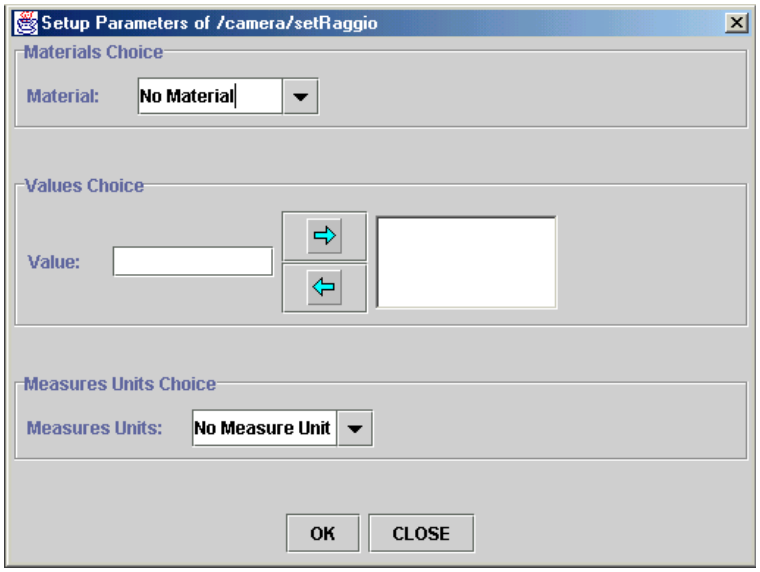


Figura 43: Modifica parametri del comando selezionato.

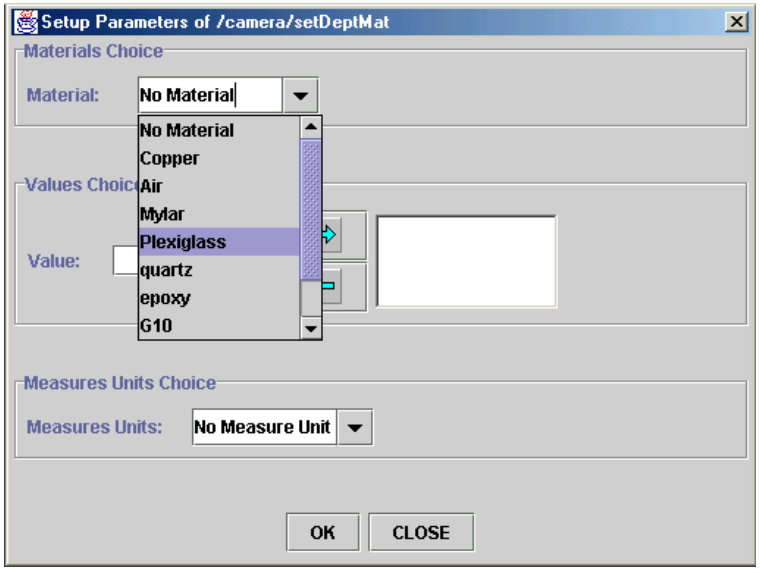


Figura 44: Selezione del materiale.

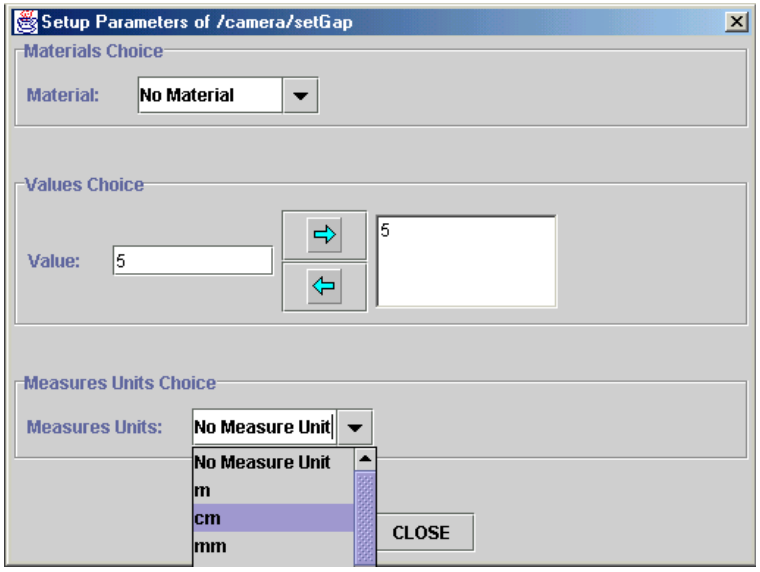


Figura 45: Selezione valore/i e unità di misura.

L'esecuzione della simulazione viene effettuata selezionando il menu 'Run > Start'. Il sistema richiede la scelta dell'applicazione da eseguire e dopo la conferma il programma viene avviato con la macro precedentemente costruita. I risultati vengono visualizzati nell'area *Console Level* come mostra la figura (47).

.4 Conclusione

Il vantaggio principale di avere un'interfaccia grafica per la scrittura e la modifica di file macro è quello di facilitare e velocizzare le operazioni di configurazione.
In futuro questa interfaccia potrà essere estesa con nuove funzionalità, in particolare può essere utile implementare un modulo per la visualizzazione di grafici e istogrammi prodotti dalle simulazioni, così da poter esportare i risultati su qualsiasi piattaforma.
Inoltre Java è stato studiato anche per applicazioni Internet, quindi l'interfaccia può essere facilmente adattata per essere eseguita sul Web, attraverso un qualsiasi browser.

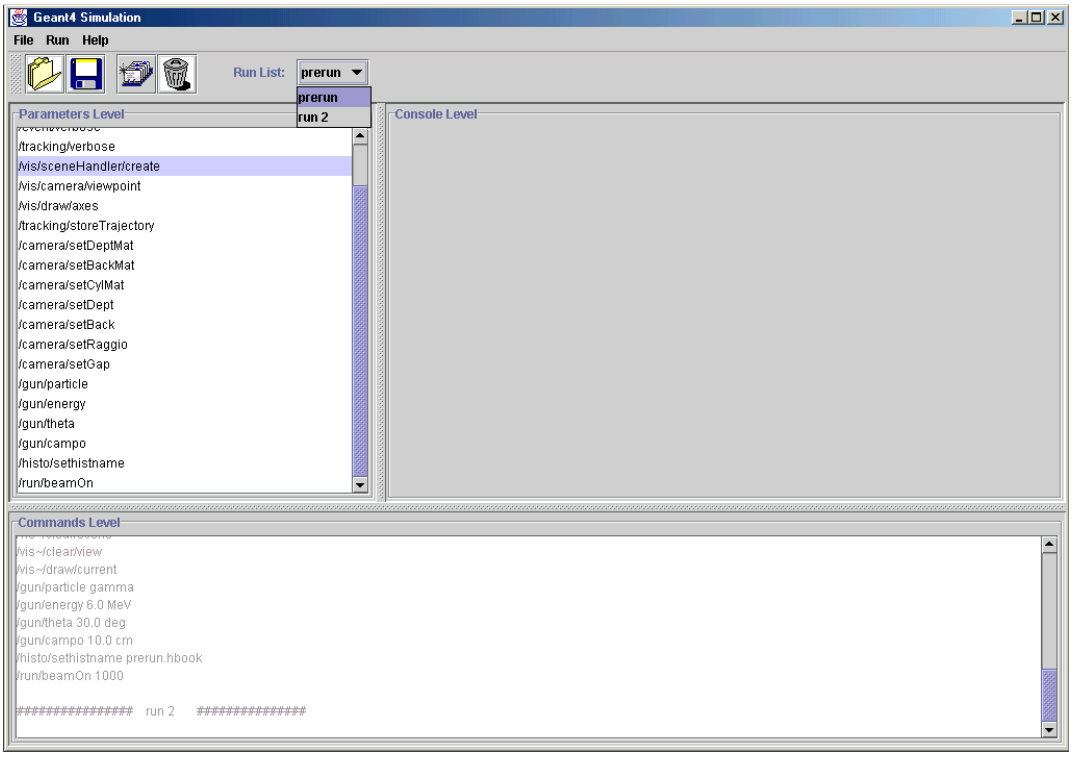


Figura 46: Selezione del run.

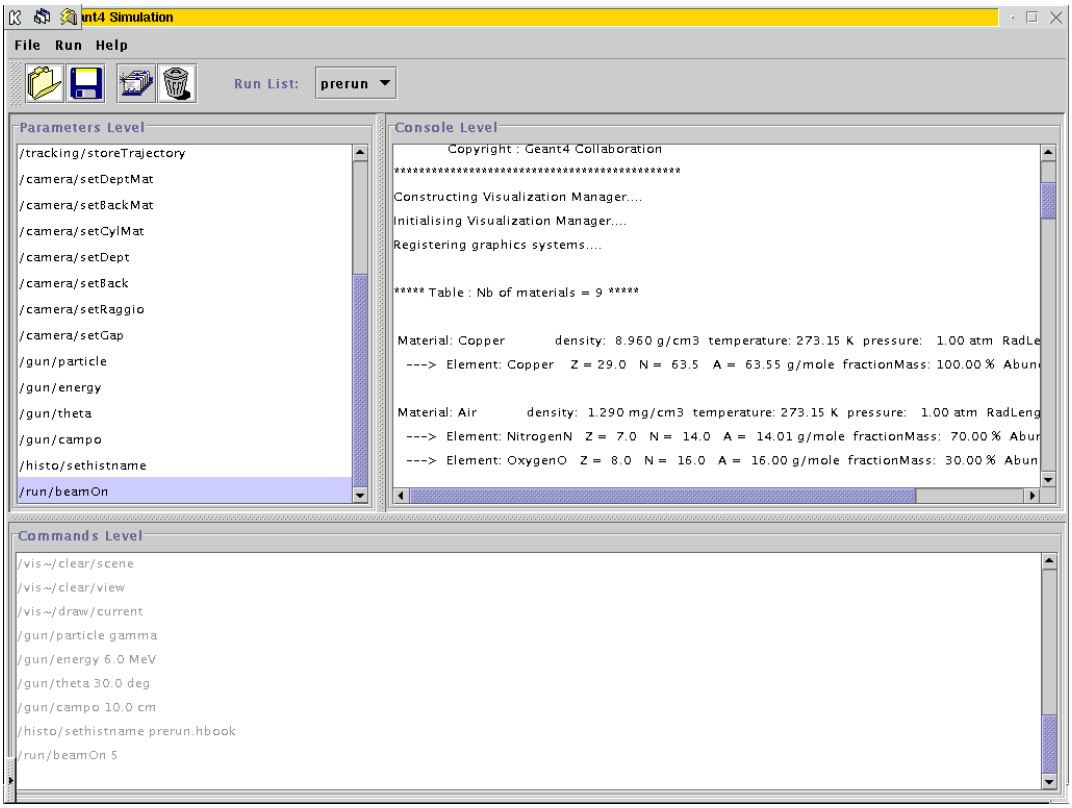


Figura 47: Esecuzione della simulazione e visualizzazione dell'output.

Bibliografia

[1] Emilio Segrè
Nuclei e particelle
Zanichelli

[2] R. Brun
Physics Reference Manual
http://www info.cern.ch/asd/geant4/G4UserDocuments/
CERN (1999)

[3] T. Bressani
Appunti di Fisica Superiore
Torino (2000)

[4] Enciclopedia Bompiani
volumi 24-25-26
Grolier Italia (1994)

[5] SL 75-5
Linear Accelerator
Philips Electronics U. K. Limited (1996)

[6] J. E. Coggle
Effetti biologici delle radiazioni
Edizioni Minerva Medica (1985)

[7] U. Amaldi
Il centro nazionale di adroterapia oncologica a Mirasole
INFN Ufficio Pubblicazioni

[8] *Introduction to Geant4*
http://www.info.cern.ch/asd/geant4/
CERN Ginevra (dicembre 2000)

[9] *Geant4 User's Guide _ For Toolkit Developers*
http://www.info.cern.ch/asd/geant4/G4UserD....ToolkitDevelopers/
CERN Ginevra (dicembre 2000)

[10] *Geant4 User's Guide _ For Application Developers*
http://www.info.cern.ch/asd/geant4/G4UserD...ApplicationDevelopers/
CERN Ginevra (dicembre 2000)

[11] k.Amako
Methodologies and Tools
http://www.info.cern.ch/asd/geant/geant4_public/pub_methodology.html
CERN Ginevra (agosto 1995)

[12] Geant4 Collaboration
RD44_Collaborating Institutes
http://www.info.cern.ch/asd/geant/geant4_public/map/sites.html
CERN Ginevra (novembre 1997)

[13] GF
Object Oriented _ Analysis and Design
http://www.info.cern.ch/asd/geant4/OOAandD/
CERN Ginevra (novembre 1999)

[14] M. Gallio e E. Chiavassa
Appunti di Fisica Nucleare
Torino (maggio 2000)