**UNIVERSITÀ DEGLI STUDI DELLA BASILICATA**

DIPARTIMENTO DI INGEGNERIA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA E DELLE TECNOLOGIE DELL'INFORMAZIONE

**RELAZIONE PROGETTO**

Sensori, Rivelatori e Dispositivi Elettronici

Docente:

Prof. Antonio Iula

Studente:

Michael Pio Stolfi 68787

**ANNO ACCADEMICO 2024-2025**

Sommario

[Sommario 2](#_Toc218613666)

[1. Introduzione 3](#_Toc218613667)

[2. Il metodo CCM 4](#_Toc218613668)

[2.1. Dati di partenza 4](#_Toc218613669)

[2.2. Ridimensionamento 5](#_Toc218613673)

[2.3. Estrapolazione 6](#_Toc218613674)

[2.4. Binarizzazione 9](#_Toc218613675)

[2.4.1. Stima della profondità utile di binarizzazione 9](#_Toc218613682)

# Introduzione

Lo scopo del presente documento è quello di riassumere, analizzare e presentare in maniera composta e ordinata i risultati ottenibili con il “*metodo CCM*”.

Il metodo denominato CCM (dai suoi creatori Capece, Caporale, Manfreda), è un metodo di feature extraction utile nell’estrazione del pattern venoso del palmo di una mano. Insieme con il presente documento è presente anche il manuale originale del metodo, che però in alcuni punti, risulta poco chiaro e oltremodo ostico. Uno degli obbiettivi di questo elaborato sarà proprio la spiegazione dettagliata con immagini a supporto del funzionamento del metodo e dei suoi effetti pratici sul volume da elaborare.

Nello specifico ci si concentrerà sul passaggio di binarizzazione, che rappresenta il cuore del metodo CCM nonché la sua parte più complessa.

# Il metodo CCM

## Dati di partenza

Il metodo CCM prende come input inziale i file “.mat” contenenti i dati relativi alle varie acquisizioni dei vari utenti. È possibile osservare il contenuto di un file “.mat” nella cattura seguente:

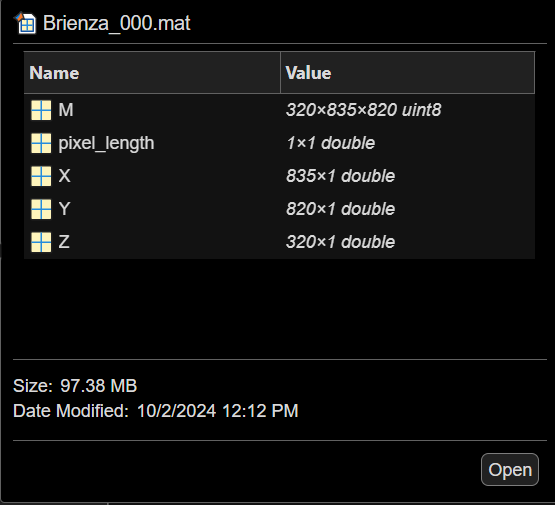


Figura 1: Preview MATLAB di un file .mat relativo all’acquisizione "00" dell'utente "Brienza".

Delle matrici precedenti quella di interesse è la matrice M, che come è possibile osservare è una matrice tridimensionale di dimensioni originali 835×820×320 voxel. Per tale matrice, come è possibile notare anche dalle dimensioni dei singoli vettori colonna, la convenzione utilizzata per le dimensioni è la seguente:

* 1ª dimensione (i) → righe → contiene i dati dell’asse Z;
* 2ª dimensione (j) → colonne → contiene i dati dell’asse X;
* 3ª dimensione (k) → pagine → contiene i dati dell’asse Y.

Quindi la matrice di partenza è nella forma zxy, però molte funzioni di elaborazione e stampa delle immagini tridimensionali in MATLAB usano la forma yxz. Il passaggio dalla prima alla seconda forma è banale infatti basta eseguire il seguente pezzo di codice:

La precedente matrice può essere ora comodamente mostrata in MATLAB grazie all’ausilio della funzione predefinita “”. Per mostrare questo volume e tutti quelli seguenti verrà utilizzata la funzione custom “*graficoVolshow(…)*” che di occupa in autonomia della creazione di alcuni componenti grafici necessari a lanciare . La matrice tridimensionale di partenza si presenta quindi come segue:

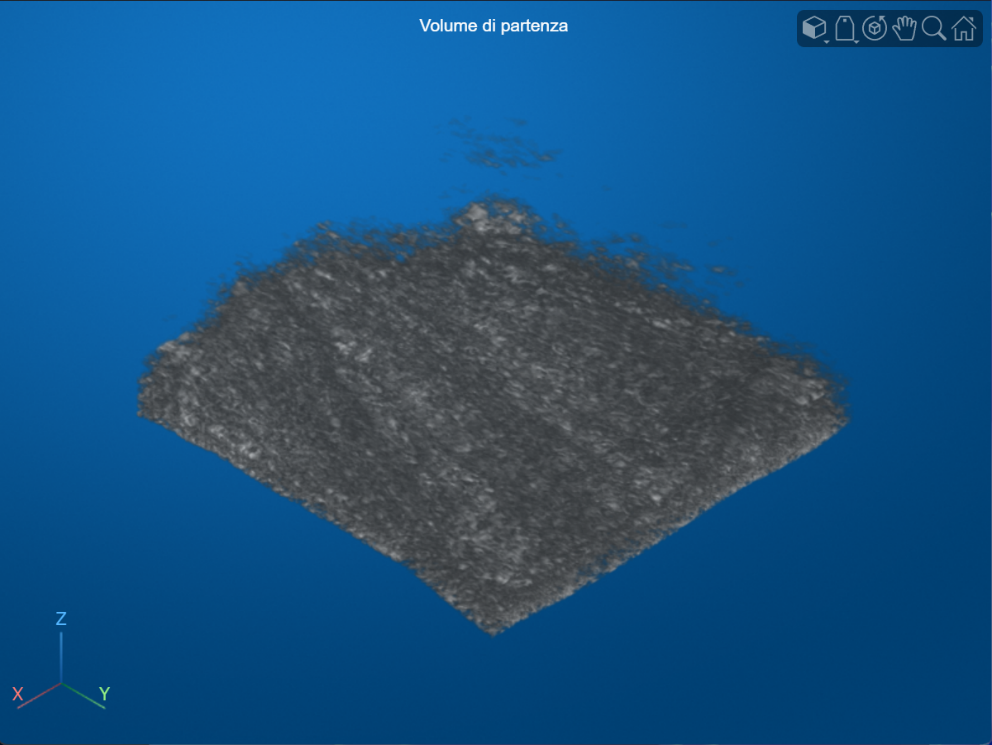


Figura 2: Rappresentazione grafica della matrice M relativa all’acquisizione "00" dell'utente "Brienza".



## Ridimensionamento

La funzione ***cropMatrice.m*** ridimensiona la matrice M, di dimensioni originali 835×820×320 voxel, estraendone una porzione centrale di 650×650×320 voxel.

La zona centrale del volume contiene le informazioni più rilevanti per l’analisi, poiché qui le vene risultano maggiormente visibili e distinguibili. Al contrario, i bordi del volume presentano un’elevata quantità di "rumore" e artefatti indesiderati che complicano il processo di feature extraction; questi disturbi, essendo troppo attaccati alle vene, sono difficili da separare senza compromettere la qualità d’insieme del pattern venoso. Il cropping centrale permette di isolare la parte interna della matrice, riducendo l'influenza del rumore periferico e preservando dettagli rilevanti per l’estrazione delle feature, senza "sporco" eccessivo.

Inoltre, la riduzione della matrice M a una risoluzione più contenuta riduce significativamente il numero di voxel da elaborare, con un conseguente risparmio nei tempi di calcolo. Questo è particolarmente vantaggioso nelle successive fasi di estrazione delle feature, che altrimenti richiederebbero un elevato carico computazionale per l’analisi dell’intero volume originario.

Di seguito è possibile osservare sempre la precedente matrice M ridimensionata come appena descritto:

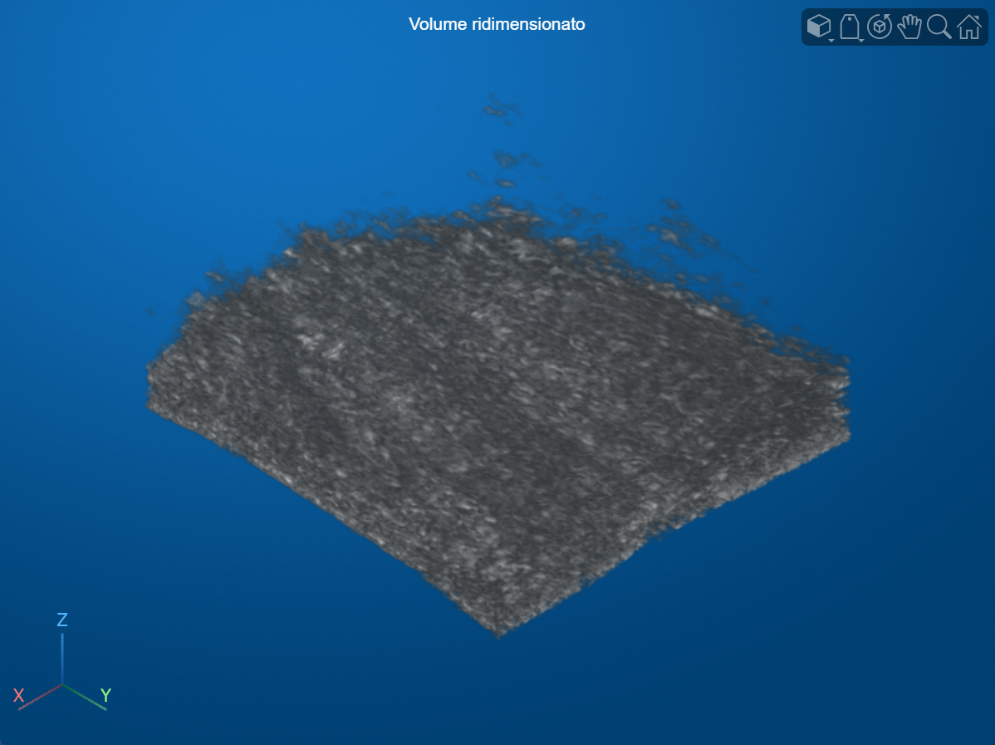


Figura 3: Rappresentazione grafica della solita matrice M dopo il ridimensionamento (ovvero Mc).

## Estrapolazione

La funzione ***estrapolaVolumeVene.m*** ha lo scopo di isolare, a partire dal volume M, la porzione realmente utile del palmo in cui ricercare il pattern venoso, eliminando le componenti non informative (acqua/rumore e regioni nere) e producendo anche maschere di supporto per i passaggi successivi. Per prima cosa stima la superficie del palmo usando una soglia di intensità (): costruisce un volume di indici di profondità coerente con la convenzione dell’acquisizione (tramite flip), azzera tali indici dove l’intensità è sotto soglia e, prendendo il massimo lungo la profondità, ottiene una mappa 2D che per ogni coordinata (x, y) identifica il voxel sopra-soglia più superficiale, cioè la quota della superficie del palmo. Questa superficie viene poi regolarizzata con un filtro passa-basso (media 20×20) per ridurre rumore e discontinuità.

Successivamente la funzione deve trasformare due quantità che hanno un significato “fisico” (espresse in millimetri) in quantità “discrete” compatibili con un volume campionato a slice. Il volume 3D infatti non è continuo: lungo l’asse di profondità (z) è composto da un certo numero di piani, e la distanza reale tra due slice consecutive non è 1 mm “per definizione”, ma dipende da come è stata fatta l’acquisizione. Questa informazione è contenuta nel vettore Z (anch’esso presente come visto prima nel solito file mat), che rappresenta le profondità (in mm) associate alle slice: in pratica indica a quale profondità reale corrisponde la k-esima slice. Di conseguenza, la differenza fornisce il passo di campionamento lungo z, cioè quanti millimetri separano due slice adiacenti.

A questo punto la funzione prende due parametri: e . Il parametro (0.2 mm) indica di quanto si vuole spostare la superficie stimata verso l’interno (o comunque lungo la profondità) per scegliere una quota di riferimento più adatta all’analisi: non si lavora esattamente sulla superficie stimata, ma su un piano leggermente traslato, perché la zona immediatamente superficiale può essere più rumorosa o meno informativa. Il parametro (0.2 mm) rappresenta invece lo “spessore” della regione che si vuole considerare attorno a quella quota, cioè una fascia in profondità che non è un singolo piano, ma un intervallo di slice.

Per poter applicare queste scelte in un volume indicizzato, la funzione converte i millimetri in numero di slice. Lo fa dividendo ciascuna distanza per il passo e arrotondando: dice quante slice corrispondono a 0.2 mm, mentre dice quante slice servono per coprire 0.2 mm di spessore. Una volta ottenuti questi indici, la funzione può definire in modo coerente la “fascia” attorno alla superficie stimata: prima sposta la superficie filtrata di slice per posizionarsi alla profondità desiderata, poi usa per stabilire quanto ampia deve essere la zona di transizione (la rampa) e quindi quali voxel devono essere considerati sopra la fascia (da scartare), dentro la fascia (transizione graduale) e sotto la fascia (da mantenere). In questo modo la selezione della regione in profondità rimane espressa in millimetri (quindi interpretabile fisicamente), ma viene applicata correttamente sui dati discreti del volume 3D.

A questo punto costruisce una maschera tridimensionale A con valori tra 0 e 255: i voxel al di sopra della fascia vengono posti a 0 (da scartare), quelli al di sotto a 255 (da mantenere) e, se abilitata la trasparenza, viene applicata una rampa intermedia per ottenere una transizione morbida nello spessore considerato. Da A ricava la mascheraAcqua (regioni sotto-soglia) e azzera tali voxel nel volume, rimuovendo quindi acqua e rumore.

Il volume e la maschera vengono poi ri-orientati per essere compatibili con la convenzione usata dal resto della pipeline (inversione dell’asse di profondità e permutazione delle dimensioni per ottenere l’ordine [y, x, z]). Di seguito è possibile apprezzare una visualizzazione del volume ripulito dall’acqua.

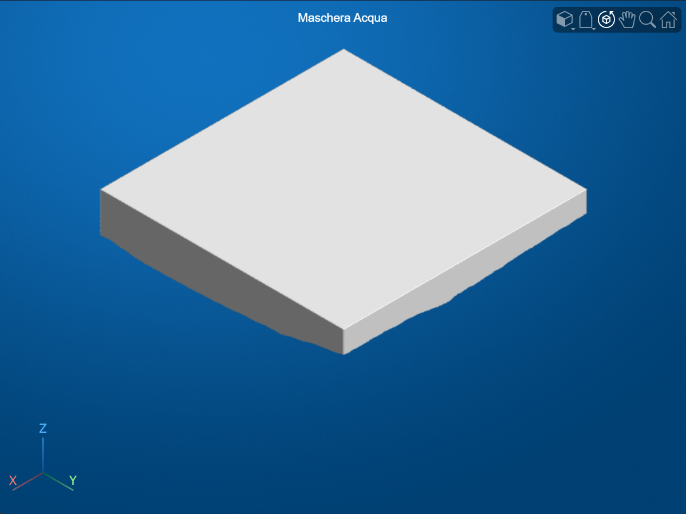
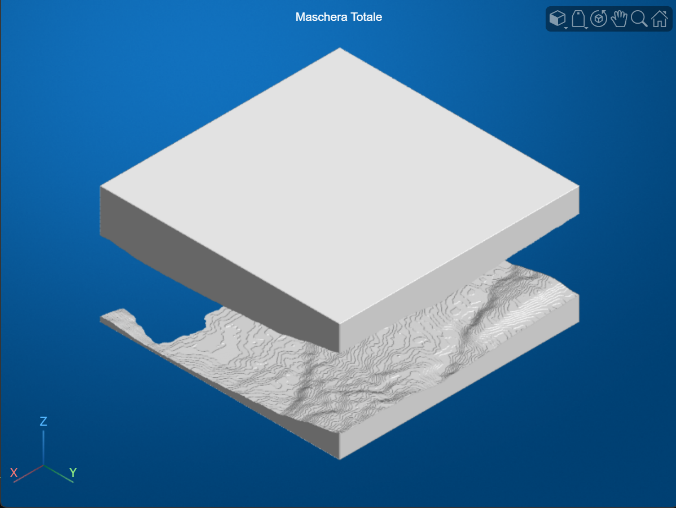
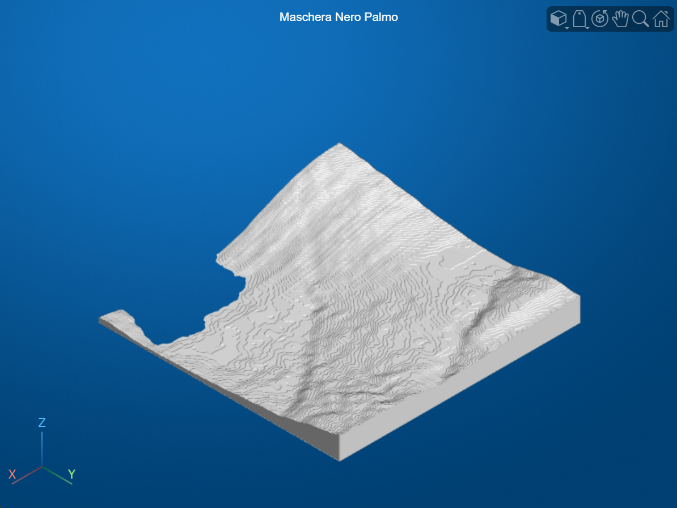


Figura 4.a: Maschera volta all’eliminazione dell’acqua; Figura 4.b: Rappresentazione grafica della matrice Mc dopo la rimozione dell'acqua (ovvero Mstart).

In seguito, tramite la funzione ***calcolaMaschere.m***, vengono calcolate le maschere delle regioni nere e viene ricavato , cioè il volume che rappresenta la regione del palmo effettivamente utilizzabile. Combinando la maschera del nero del palmo con la maschera dell’acqua, la funzione produce anche , una versione del volume in cui sono stati rimossi sia acqua/rumore sia le porzioni nere del palmo. Di seguito è possibile apprezzare una visualizzazione del volume ulteriormente ripulito.



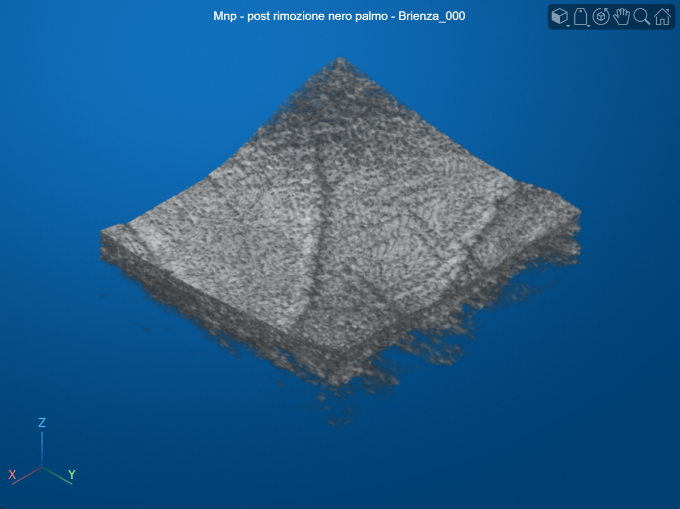


Figura 5.a: Maschera volta all’eliminazione del nero del palmo; Figura 5.b: Maschera totale che alla fine è stata applicata sul volume di partenza; Figura 5.c: Rappresentazione grafica della matrice Mnp dopo la rimozione del nero del palmo.

Infine, per ogni coordinata (x, y), la funzione calcola l’indice della prima slice lungo z in cui risulta non nullo, ottenendo una mappa della “quota di inizio palmo” (). Sottraendo un offset fisso () restituisce indiciPalmoNoPelle, che rappresenta una stima dell’inizio del palmo al netto dello strato più superficiale (pelle), utile per concentrare l’analisi su profondità più informative per il pattern venoso.

## Binarizzazione

La funzione ***effettuaBinarizzazione.m*** realizza l’intera pipeline di **binarizzazione 3D** finalizzata all’estrazione delle **strutture venose** dal volume del palmo. L’input principale è il volume pre-processato ***Mnp*** (con intensità in 0–255), accompagnato dalle maschere di esclusione (mascheraAcqua, mascheraNeroTotale) e dalla mappa di superficie del palmo senza pelle indiciPalmoNoPelle, utilizzata per riferire le operazioni alla geometria del palmo lungo la profondità. In uscita produce: (i) ***volBinFinal***, ossia il volume binario finale delle vene, e (ii) ***vecProcessed***, un vettore “smussato” che descrive, per ciascun piano y, fino a quale profondità conviene binarizzare. Tale funzione si compone di diversi passi:

1. Calcolo del vettore fine binarizzazione;
2. Binarizzazione iniziale e pulizia morfologica;
3. Stima distanze vene–palmo e stima dei diametri;
4. Stima automatica della soglia iniziale;
5. Binarizzazione finale (incrementale e vincolata);
6. Salvataggio su disco.

Ognuno dei passi precedenti, essendo tutti piuttosto complicati, è effettuato da una o più funzioni dedicate.



### Calcolo del vettore fine binarizzazione



La funzione ***calcolaVecFineBin.m*** costruisce un vettore (lungo ) che, per ogni slice lungo y del volume, stima un valore ottimale del parametro **fine** usato poi nella binarizzazione dei piani XZ: in pratica indica fino a che profondità sotto la superficie del palmo conviene “scendere” quando si binarizza quel piano.

Per prima cosa legge le dimensioni del volume assumendo la convenzione (y, x, z), crea un volume a contrasto invertito (così strutture originariamente scure diventano ad alta intensità) e azzera tutte le zone non utili usando . Fissa inoltre , cioè il limite inferiore dell’intervallo di profondità considerato nella binarizzazione.

La stima di fine avviene poi con una scansione controllata: dalla maschera acqua ricava, per ogni coppia (y, x), l’indice z del primo voxel marcato come acqua tramite e impone un limite globale conservativo , cioè non prova mai valori di fine oltre una profondità che potrebbe già ricadere in acqua in qualche punto del volume. A questo punto, per ogni y, estrae il piano XZ del volume invertito come (porta il piano nella forma attesa dalle funzioni successive), prende il profilo della superficie “palmo senza pelle” e costruisce un vettore vecCC lungo . Per ogni candidato binarizza il piano chiamando ***binPianoSingolo.m***, converte la maschera logica risultante in un’immagine binaria uint8 a 0/255 (formato richiesto dalla funzione di conteggio) e calcola quante componenti connesse sono presenti con , salvando in . Terminata la scansione, cerca i picchi di vecCC con e sceglie l’indice ottimo del piano in modo deterministico: se i picchi sono almeno due usa la media dei primi due, se ce n’è uno solo usa il primo, se non ce ne sono assegna NaN. Quel valore viene salvato in vecFineBin(y). Finito il ciclo su tutte le y, crea la cartella di output se manca e salva vecFineBin su disco solo se contiene almeno un valore diverso da zero altrimenti evita di salvare un risultato vuoto.



Figura 6: Plot del numero delle cc per ogni iterazione di un piano XZ.

La scelta di limitarsi ai primi due picchi è del tutto arbitraria e potrebbe rivelarsi non ottima.

Una volta completati tutti i piani il vettore ricavato viene processato tramite la funzione ***processaVettore.m*** la quale restituisce il vettore .

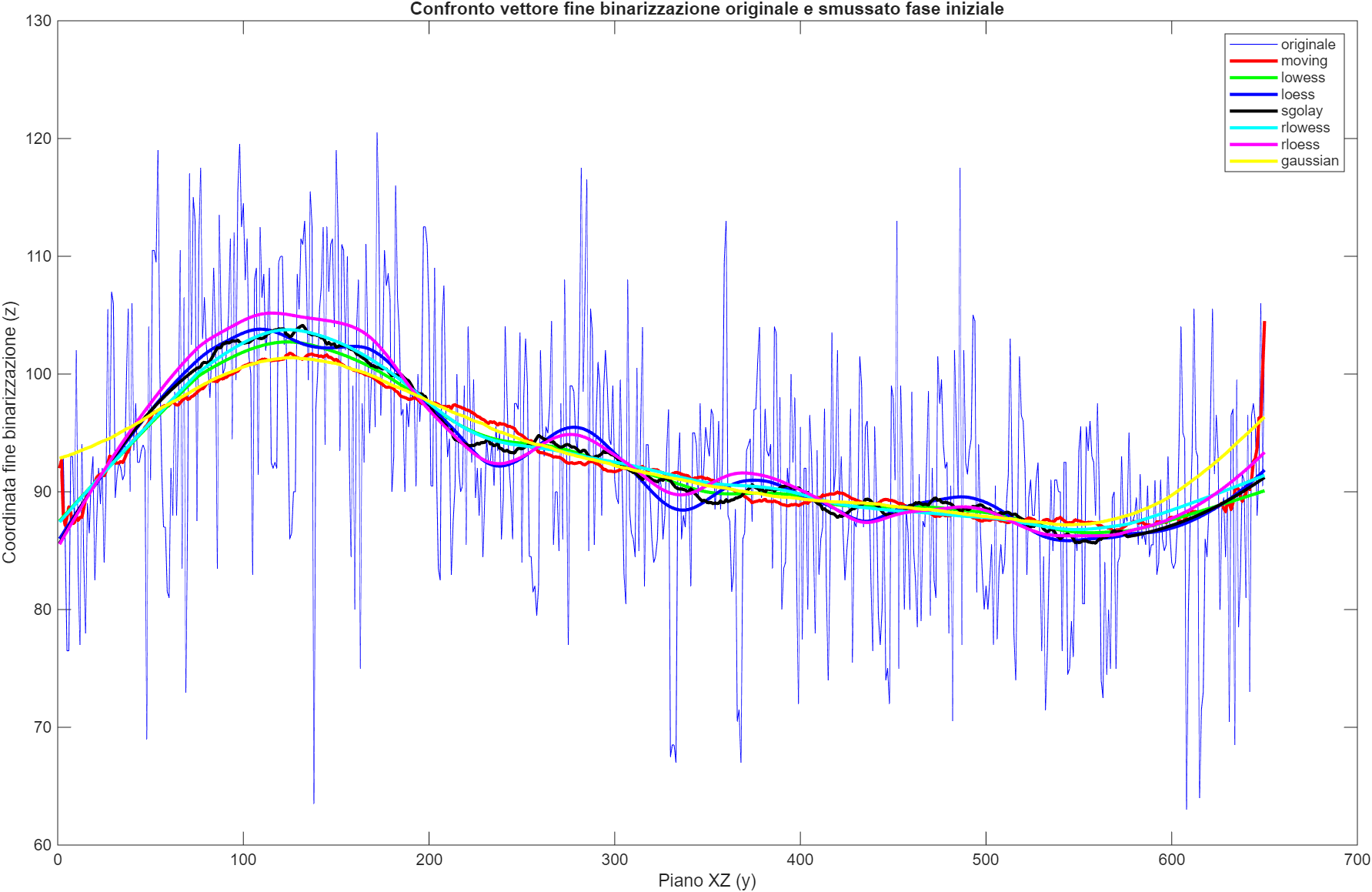


Figura 7: Fine binarizzazione calcolata per ogni piano XZ del volume.

Nella Figura 7 è possibile vedere il valore di fine ottimo calcolato per ogni piano XZ il quale rappresenterà l’andamento della binarizzazione lungo la dimensione Z e quindi sarà in grado di ricavare sia le vene più pendenti che quelle costanti. Per rendere la caratterizzazione delle vene più omogenea viene applicata al vettore in questione una procedura di smoothing. Lo smussamento va a regolare i picchi non dando peso a possibili outlier ma conservando l’andamento generale delle vene profonde. Sono stati provati diversi operatori di smoothing ma alla fine si è visto che l’operatore rlowess funziona meglio per questo scopo (tratto turchese).

Da questo vettore possiamo intuire come l’utente in questione abbia nella zona del BSCAN che va da 0 a 200 la vena principale ad una profondità che parte nell’intorno di 90 per poi scendere in profondità fino a 100 e infine risalire più verso la superficie da 200 in poi.

### Binarizzazione iniziale e pulizia morfologica

La funzione ***binarizzaVolume.m*** esegue la binarizzazione 3D del volume lavorando piano per piano, sui piani XZ (uno per ogni indice y). Nel contesto di questo step “iniziale” viene invocata con e , condizione che attiva la modalità di binarizzazione preliminare (non incrementale) pensata per ottenere una prima estrazione “grezza ma affidabile” delle vene.

All’inizio la funzione prepara un volume di intensità adatto alla soglia: calcola , cioè inverte il contrasto, e impone a zero tutte le regioni non utilizzabili tramite . L’inversione è importante perché rende “alte” (quindi facilmente sogliabili) le strutture che nell’immagine originale sono più scure; la maschera invece elimina a monte zone nere/acqua o comunque non affidabili, impedendo che generino falsi positivi. Viene poi allocato un volume binario finale (tutto falso) delle stesse dimensioni del volume originale (convenzione y, x, z), che verrà riempito progressivamente.

La binarizzazione vera e propria è eseguita indipendentemente per ogni y (in parallelo). Per ciascun piano, viene letto il parametro , che rappresenta fino a quante slice di profondità sotto la superficie conviene analizzare in quel piano. A questo punto entra in gioco la mappa , cioè l’indice z della superficie del palmo (senza pelle) per ogni colonna x: la soglia non viene applicata su una profondità assoluta fissa, ma su una fascia di profondità relativa alla superficie locale. In pratica, per ogni profondità relativa compresa tra 1 e fine, e per ogni x, si campiona il voxel situato a quota (se l’indice resta dentro il volume) e lo si marca come appartenente a vena se l’intensità invertita supera una soglia molto alta. In questa fase la soglia usata è 250 su Minv, quindi vengono selezionati solo voxel “quasi saturi” dopo inversione: equivalgono, nell’immagine originale, a intensità molto basse (strutture molto scure), scelta coerente con una binarizzazione iniziale volutamente conservativa.

Il risultato ottenuto con questa prima soglia è poi raffinato con un passaggio aggiuntivo (***binOffsetVena.m***), che serve a “ritagliare” e rendere più coerente la regione lungo z in cui mantenere le attivazioni. In particolare, sul piano XZ binarizzato vengono calcolate le componenti connesse 2D e i relativi centroidi; viene individuata la componente con centroide più “superficiale” (minimo in z) e, a partire da quella quota, la maschera viene limitata/propagata mantenendo solo le attivazioni entro un offset fisso di 60 slice verso profondità maggiori. Operativamente, questo passaggio tende a scartare attivazioni troppo superficiali o isolate e a concentrare la maschera nella fascia in cui la vena risulta più plausibile e consistente nel piano. Il piano così ottenuto viene poi riportato nel volume 3D finale (riallineando correttamente gli assi del piano rispetto alla convenzione y, x, z). L’output di questa modalità è quindi un volume binario preliminare **volBin** che rappresenta una prima estrazione delle vene, costruita (i) rispettando la geometria del palmo tramite indiciPalmoNoPelle, (ii) limitando la profondità con vecFine, e (iii) usando una soglia molto selettiva per ridurre rumore e falsi positivi. Il risultato ottenuto è visibile nell’immagine sottostante:

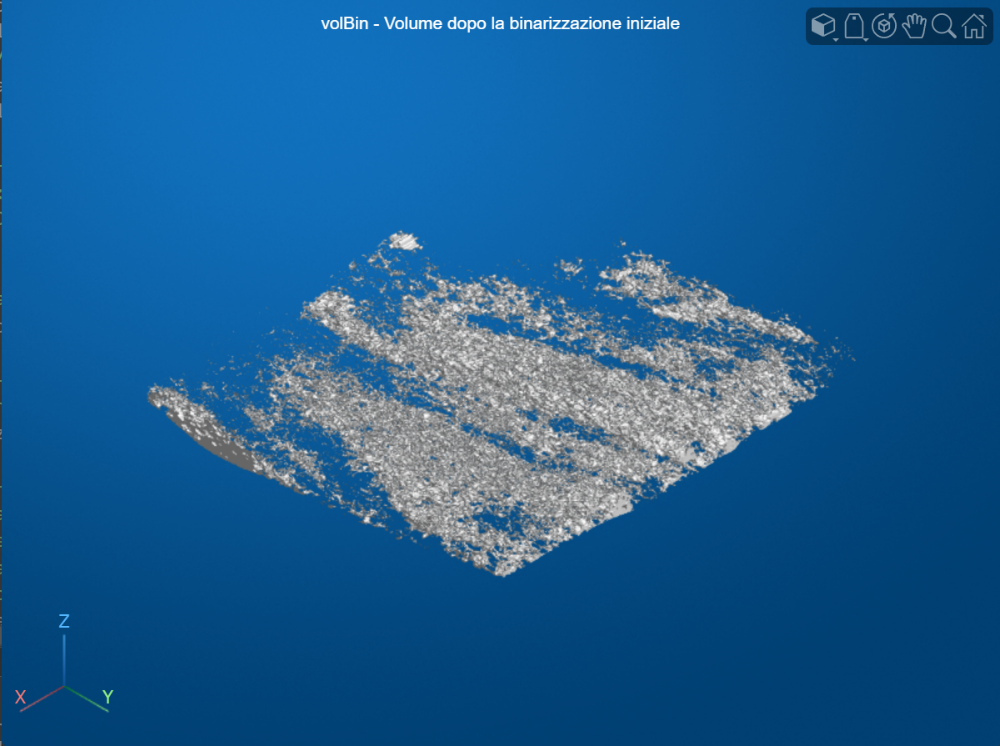


Figura 8: Rappresentazione grafica della matrice volBin dopo una prima binarizzazione iniziale.

La funzione ***separaStrutture.m*** prende in ingresso il volume binario prodotto dalla binarizzazione iniziale e lo trasforma in un volume “pulito” e più stabile, eliminando rumore, frammenti e connessioni spurie. Il primo intervento è un filtraggio per dimensione tramite con soglia 500 voxel: tutte le componenti connesse più piccole vengono rimosse, perché tipicamente riconducibili a rumore o attivazioni isolate non venose. Dopo questo step il volume viene portato in formato 0/255 (valori utili sia per debug sia per alcune funzioni successive), mantenendo però la natura binaria dell’informazione.

Successivamente viene applicata un’apertura morfologica 3D () con elemento strutturante sferico di raggio 1. Questa operazione (erosione seguita da dilatazione) ha lo scopo di eliminare piccole sporgenze, interrompere ponti sottili e separare strutture che risultano connesse solo per contatti minimali, migliorando la separabilità degli oggetti effettivamente presenti. A valle dell’apertura, la funzione applica anche uno smoothing gaussiano 3D con sigma 1 e padding “circular”: l’effetto è quello di ammorbidire i bordi e colmare micro-discontinuità, rendendo la successiva ri-binarizzazione meno sensibile a irregolarità locali.

Dopo lo smoothing, il volume viene nuovamente binarizzato con una soglia manuale pari a 100 (cioè si torna a una rappresentazione netta dopo aver reso le strutture più regolari). Infine viene eseguita una seconda pulizia più severa: un ulteriore mantiene soltanto le componenti connesse con volume almeno 2000 voxel, eliminando i residui rimasti dopo la fase di smoothing e ri-soglia. In termini funzionali, separaStrutture produce dunque un volume binario molto più consistente e robusto, che viene poi utilizzato negli step successivi (ad esempio stima distanza vene–palmo e diametri), dove è fondamentale che le componenti siano sufficientemente grandi e coerenti per estrarre misure geometriche affidabili. Il risultato ottenuto è mostrato nella figura seguente:



Figura 9: Rappresentazione grafica della matrice volumeSeparato dopo una prima serie di operazioni morfologiche atte a separare le strutture dal rumore.

### Stima distanze vene–palmo e stima dei diametri

Dopo la binarizzazione iniziale e la pulizia morfologica (che producono un volume di vene già abbastanza coerente), la funzione ***calcolaMinDistVenePalmo.m*** ha il compito di ricavare, per ogni piano XZ (cioè per ogni indice y), una misura geometrica fondamentale: la distanza minima tra le vene segmentate e il palmo lungo la direzione z. In parallelo tenta anche di stimare, sullo stesso piano, un diametro plausibile della vena (quando possibile). Queste due informazioni diventano poi vincoli essenziali per la binarizzazione “finale”: la distanza aiuta a capire fino a che profondità conviene cercare le vene, mentre il diametro serve come parametro geometrico per rendere più robusta l’estrazione.

La funzione inizia con un taglio “grossolano” dei bordi: azzera manualmente due fasce laterali lungo x (da 1 a 50 e da 600 a 650) sia nel volume vene sia nel volume palmo. Questo step è un filtro empirico per eliminare regioni esterne che, per caratteristiche del sensore o della porzione acquisita, risultano frequentemente rumorose o poco affidabili. Subito dopo applica un ulteriore taglio mediante ***tagliaBordi.m***, che rimuove porzioni agli angoli (in modo più “geometrico” e uniforme su tutte le slice), riducendo ancora la probabilità che artefatti periferici influenzino misure di distanza e diametro. A scopo di controllo visivo viene anche costruito un volume unito (OR logico tra palmo e vene) e mostrato con ***graficoVolshow***, così da verificare che le vene risultino effettivamente “sotto” il palmo e che i tagli non abbiano eliminato regioni di interesse.

Il calcolo vero e proprio avviene poi piano per piano (loop parallelo su y). Per ogni y, la funzione estrae il piano XZ delle vene e quello del palmo, e applica al piano vene una piccola pulizia 2D: un’erosione con elemento strutturante sferico di raggio 1 e un riempimento dei buchi (). Questo serve a stabilizzare le regioni venose nel singolo piano, eliminando pixel isolati e rendendo le strutture più compatte, così che il successivo conteggio/filtraggio per componenti sia più affidabile. Successivamente calcola le componenti connesse del piano venoso e seleziona solo quelle con area compresa tra 20 e 2500 pixel: l’obiettivo è scartare sia rumore (oggetti troppo piccoli) sia regioni troppo grandi (che difficilmente rappresentano una singola vena plausibile o che potrebbero essere aggregazioni spurie). Il risultato di questo filtraggio è una maschera di “vene valide” del piano.

A questo punto la funzione prova anche a stimare un diametro tramite ***calcolaDiametroVena.m***, usando la maschera delle componenti valide e la geometria del palmo nel piano. Il diametro può risultare se non ci sono condizioni sufficientemente affidabili per stimarlo (ad esempio assenza di strutture adeguate o geometria non coerente). In parallelo viene calcolata la distanza vene↔palmo: per ogni colonna x del piano, si cerca nel palmo l’ultimo voxel non nullo lungo z (interpretato come “quota più profonda” del palmo in quella colonna) e, nella maschera vene valide, si cercano gli ultimi 3 voxel non nulli lungo z (cioè la parte più “profonda” della vena, imponendo però un requisito minimo di consistenza per evitare falsi positivi dovuti a un singolo voxel isolato). Solo se in quella colonna sono presenti almeno 3 voxel venosi, la distanza viene considerata misurabile e viene calcolata come (distanza in voxel lungo z). Ripetendo questa operazione su tutte le colonne x, la funzione ricava un vettore di distanze per il piano e ne salva la minima: questa scelta identifica, per quel piano y, il caso in cui la vena risulta più “vicina” al palmo (misura conservativa utile per limitare la profondità di ricerca nelle fasi successive). Se per quel piano esiste anche un diametro valido, la funzione memorizza sia il diametro sia la distanza associata (che, in questa implementazione, coincide con la minima distanza del piano). In uscita produce quindi: ***vettoreDistanze*** (una distanza minima per ogni y) e ***matriceDiametri*** (2×, con diametri stimati e distanze associate).

La funzione ***processaVettore.m*** viene usata subito dopo per trasformare un vettore grezzo calcolato per piani (come **vecDist**) in un profilo **continuo, stabile e utilizzabile** nelle fasi successive. In pratica, questi vettori possono contenere (piani in cui la misura non è stimabile) e oscillazioni locali dovute a rumore o a segmentazioni irregolari: prima di usarli come vincoli nella binarizzazione finale, conviene renderli “regolari” e senza buchi.

Il primo passaggio è la gestione dei valori mancanti: la funzione interpola linearmente i usando solo i campioni validi. Poiché l’interpolazione può lasciare agli estremi (tipicamente all’inizio, se le prime slice non hanno valori), viene gestito esplicitamente il caso in cui manchino i primi campioni: questi vengono riempiti copiando il primo valore disponibile, così da evitare un inizio indefinito del profilo. Una volta ottenuto un vettore completo, la funzione applica una fase di smoothing piuttosto forte (finestra pari a **150** campioni), calcolando diverse varianti di levigatura (media mobile, lowess/loess, Savitzky–Golay, Gaussian, e versioni robuste). Anche se ne calcola molte per confronto, l’uscita effettivamente selezionata è la versione robusta **rlowess**, scelta perché meno sensibile a picchi e outlier rispetto a una regressione locale standard. Infine, per garantire coerenza dimensionale con il numero di piani del volume, la funzione forza la lunghezza del risultato a : se il vettore risultasse più corto (ad esempio per effetti di shape/colonne), aggiunge campioni in coda replicando l’ultimo valore disponibile.

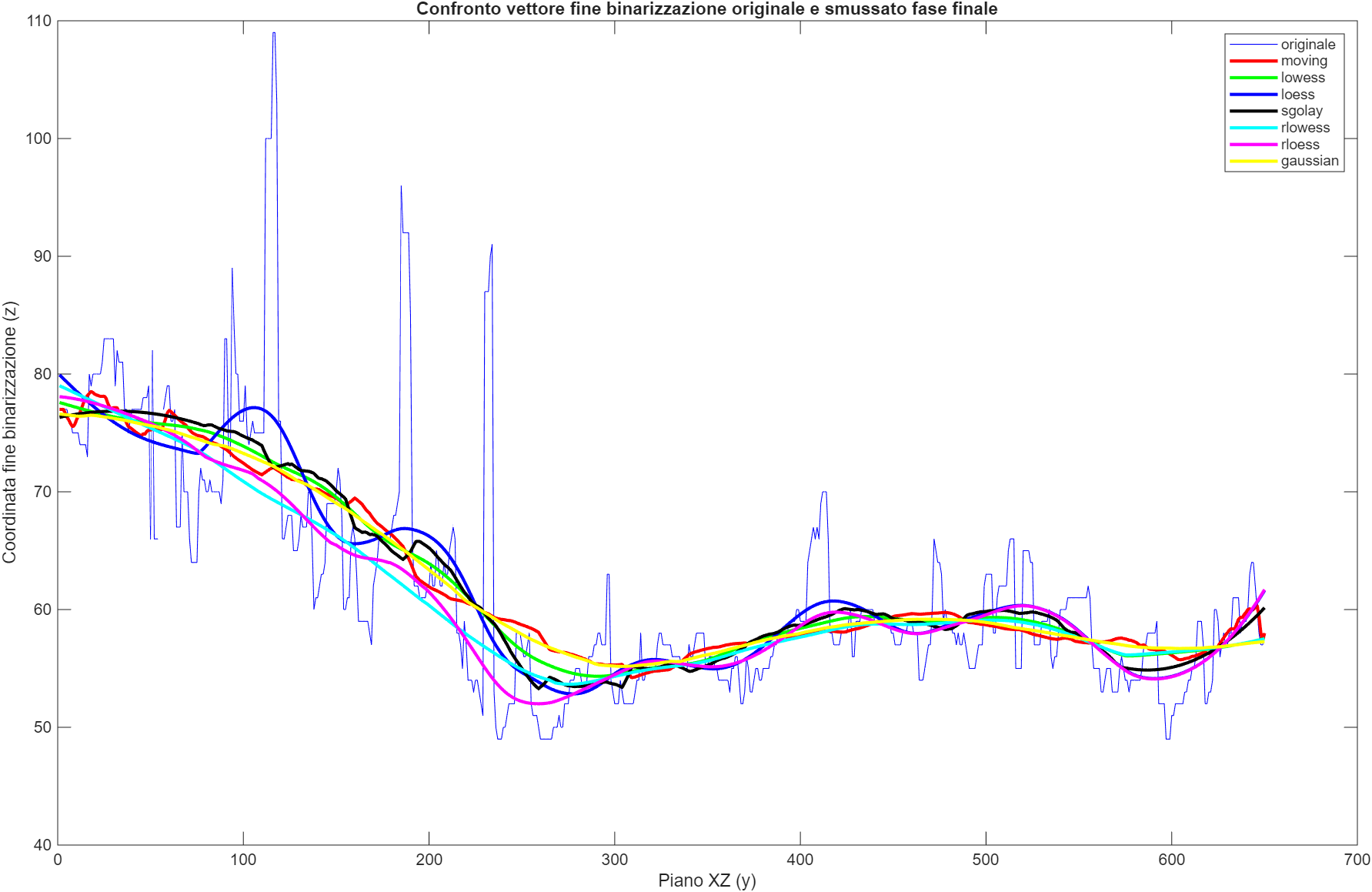


Figura 10: Distanza palmo-vena calcolata per ogni piano XZ del volume.

### Stima automatica della soglia iniziale

Dopo aver ottenuto un volume venoso “pulito” e, soprattutto, un profilo geometrico affidabile della distanza vene–palmo (), la funzione ***calcolaSogliaIniziale.m*** serve a determinare in modo automatico una soglia di intensità iniziale da cui avviare la binarizzazione finale (quella incrementale). L’idea è evitare una scelta manuale e rendere la procedura ripetibile: la soglia viene selezionata osservando come cambia la complessità del volume binarizzato al variare della soglia, misurata tramite il numero di componenti connesse “significative” (cioè non troppo piccole).

Operativamente la funzione esegue uno sweep di soglie decrescenti da 250 a 210 (step −1). Per ogni soglia candidata binarizza l’intero volume con ***binIncrementale***, imponendo però un vincolo fondamentale: la profondità massima di analisi viene fissata a un valore conservativo , cioè la minima distanza stimata tra vene e palmo lungo tutte le slice y. In questo modo, durante il confronto tra soglie, la binarizzazione avviene sempre in una fascia sicuramente compatibile con l’ipotesi “vene sotto il palmo”, evitando che profondità variabili introducano artefatti nella metrica. Dal volume binarizzato ottenuto a ciascuna soglia, vengono poi calcolate le componenti connesse 3D e, tramite ***regionprops3***, viene misurato il volume (in voxel) di ogni componente; si considerano “valide” solo quelle con e si conta quante sono. Il risultato è quindi un vettore che, per ciascuna soglia provata, contiene il numero di componenti connesse grandi emerse dalla binarizzazione.

A questo punto la scelta della soglia non viene fatta prendendo banalmente il massimo o un valore fisso, ma individuando un punto di transizione nella curva #CC(soglia). Per ridurre l’effetto del rumore dovuto alla discretizzazione, viene smussato con un fitting LOESS (finestra 10) e sulla versione smussata viene calcolata la derivata discreta , che mette in evidenza dove il numero di componenti cresce rapidamente o cambia regime. La funzione trova prima il massimo globale della curva originale (il punto in cui il numero di CC è massimo), poi cerca i massimi locali della derivata (zone di salita rapida) e i minimi locali della derivata (zone in cui la crescita rallenta/inverte). Viene quindi selezionato: (1) il massimo locale della derivata più vicino al massimo globale della curva e, (2) il minimo locale della derivata più vicino a quel massimo locale. La soglia iniziale finale è la soglia corrispondente a questo minimo locale “successivo”, cioè un punto in cui si è appena superata la fase di variazione più brusca: in termini pratici, è un compromesso che tende a evitare sia soglie troppo alte (troppo selettive, poche strutture) sia soglie troppo basse (troppo permissive, frammentazione/rumore), scegliendo un valore coerente con la dinamica osservata sul volume.

Di seguito è possibile osservare i grafici di interesse dei vettori appena discussi:

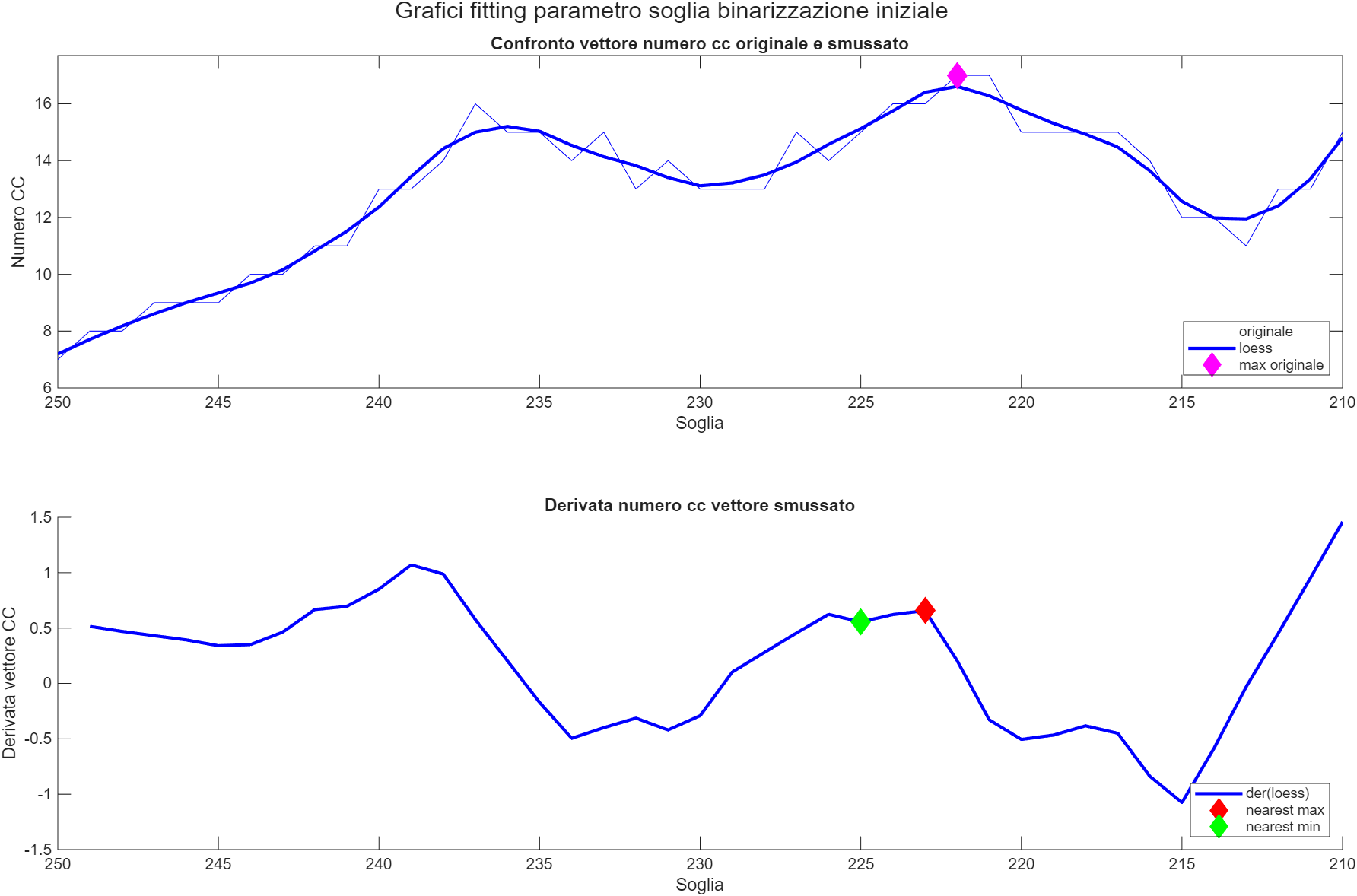


Figura 11: Plot numero cc in funzione del valore di soglia iniziale e sua derivata.

### Binarizzazione finale

Nello step finale di ***effettuaBinarizzazione***, la funzione ***binarizzaVolume.m*** viene richiamata passando come vettore , la stimata in precedenza e le soglie . In questa modalità la binarizzazione non è più “manuale” e fissa, ma diventa incrementale lungo la profondità e viene anche adattata tramite un parametro geometrico globale (la cosiddetta ), ricavato dai diametri stimati.

La funzione parte come nella fase iniziale: lavora nel riferimento del volume (y, x, z), crea per rendere alte le strutture originariamente scure e azzera tutte le regioni non utilizzabili con . Il risultato finale viene accumulato in un volume binario inizializzato a false. A questo punto, però, prima di binarizzare, viene stimato un offset di profondità da aggiungere a per rendere la ricerca più robusta e coerente con la “taglia” delle vene.

Dalla la funzione legge, per ogni piano y, un diametro stimato () e una distanza associata (). Non tutti questi diametri vengono accettati: vengono considerati “validi” solo quelli per cui è coerente con il valore misurando una differenza percentuale e imponendo che sia < 5%. Questa regola serve a evitare di usare diametri stimati su piani in cui la stima geometrica (o la segmentazione) è poco affidabile o non compatibile col profilo di profondità usato per la binarizzazione.

I diametri che superano il controllo vengono raccolti e la costante viene inizialmente posta come media arrotondata dei diametri validi (se non ce ne sono, la costante viene posta a 0). In pratica, questa costante è un parametro globale che “espande” la profondità di ricerca: se le vene hanno mediamente un certo spessore/estensione, la profondità utile non dovrebbe fermarsi esattamente a ma includere una fascia aggiuntiva.

Se la costante iniziale è > 0, viene poi raffinata automaticamente da ***calcolaCostanteDiametroVene***: qui l’idea è provare più offset (da 0 fino alla costante stimata) e scegliere quello che produce un comportamento più “informativo” nel numero di strutture venose robuste. Per ogni tentativo, la funzione binarizza il volume con ***binIncrementale*** usando come profondità e conta quante componenti connesse 3D hanno volume almeno 3000 voxel (si contano quindi solo strutture grandi, meno compatibili con rumore). Si ottiene una curva “tentativo → numero CC robuste”, la si smussa (loess), si guarda la derivata discreta e si scelgono punti di variazione (massimi/minimi locali della derivata): se i diametri validi sono molti (>20) la scelta tende a essere più “spinta” (punto più tardivo), altrimenti più conservativa (punto più precoce); se non esistono punti di variazione, si ripiega sul tentativo che massimizza direttamente il numero di CC robuste. Il risultato è una finale, più stabile rispetto alla sola media.

Definita la costante, la binarizzazione finale viene eseguita in parallelo su tutti i piani y. Per ciascun piano si imposta una profondità effettiva: , cioè si estende la profondità stimata per quel piano aggiungendo un margine legato al diametro medio delle vene. A questo punto entra in gioco ***binIncrementalePiano***, che è la funzione che realizza la binarizzazione vera e propria sul singolo piano XZ. Il principio è lo stesso già visto in altre parti della pipeline: non si soglia un’intera slice a quota z fissa, ma si lavora in coordinate relative alla superficie del palmo. Per ogni “passo” di profondità virtuale zplane da 1 a fine, viene calcolata per ogni colonna x la posizione assoluta lungo z come: . Se è valido (>0), viene campionato il voxel corrispondente nel volume invertito Minv (quindi sotto la superficie del palmo) e confrontato con una soglia. La particolarità (che giustifica il nome “incrementale”) è che la soglia non è costante, ma varia con la profondità: tra sogliaIniziale e sogliaFinale viene interpolata con un andamento quadratico in funzione di t normalizzato tra 0 e 1 (cioè la soglia cambia lentamente all’inizio e più rapidamente verso la fine dell’intervallo). In questo modo ogni voxel “candidato” viene valutato con un criterio che dipende dalla profondità a cui si trova: all’aumentare di la procedura tende a rendere la selezione progressivamente più severa (dato che qui sogliaFinale è 255). Quando un voxel supera la soglia del suo step di profondità, viene scritto come true nel volume 3D esattamente nella posizione (y, x, indice\_z). Ripetendo il processo per tutte le profondità virtuali e per tutte le colonne, il piano finale viene costruito come un insieme di voxel selezionati sotto la superficie del palmo, e ***binarizzaVolume*** lo inserisce in ***MatFinale*** come slice y del volume completo.

Il risultato di questo step è quindi un ***volBinFinal*** che nasce da tre vincoli combinati: (1) vincolo geometrico dato dalla superficie del palmo (***indiciPalmoNoPelle***), (2) vincolo di profondità dato da ***vecFine*** esteso con una costante legata al diametro venoso, e (3) criterio di selezione adattivo lungo la profondità tramite soglia incrementale. Questo volume finale è quello che ***effettuaBinarizzazione*** salva e verrà utilizzato poi per il matching/riconoscimento. Nella figura sottostante è possibile osservare come si presenta il volume dopo la binarizzazione finale:

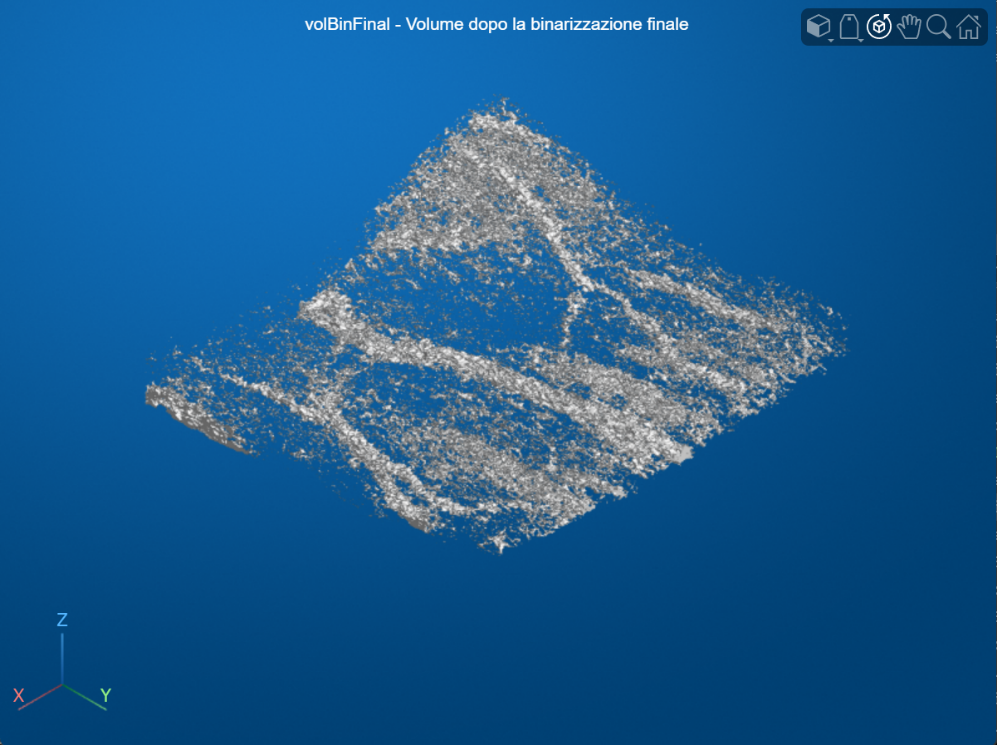


Figura 12: Rappresentazione grafica della matrice volBinFinal dopo la binarizzazione finale.