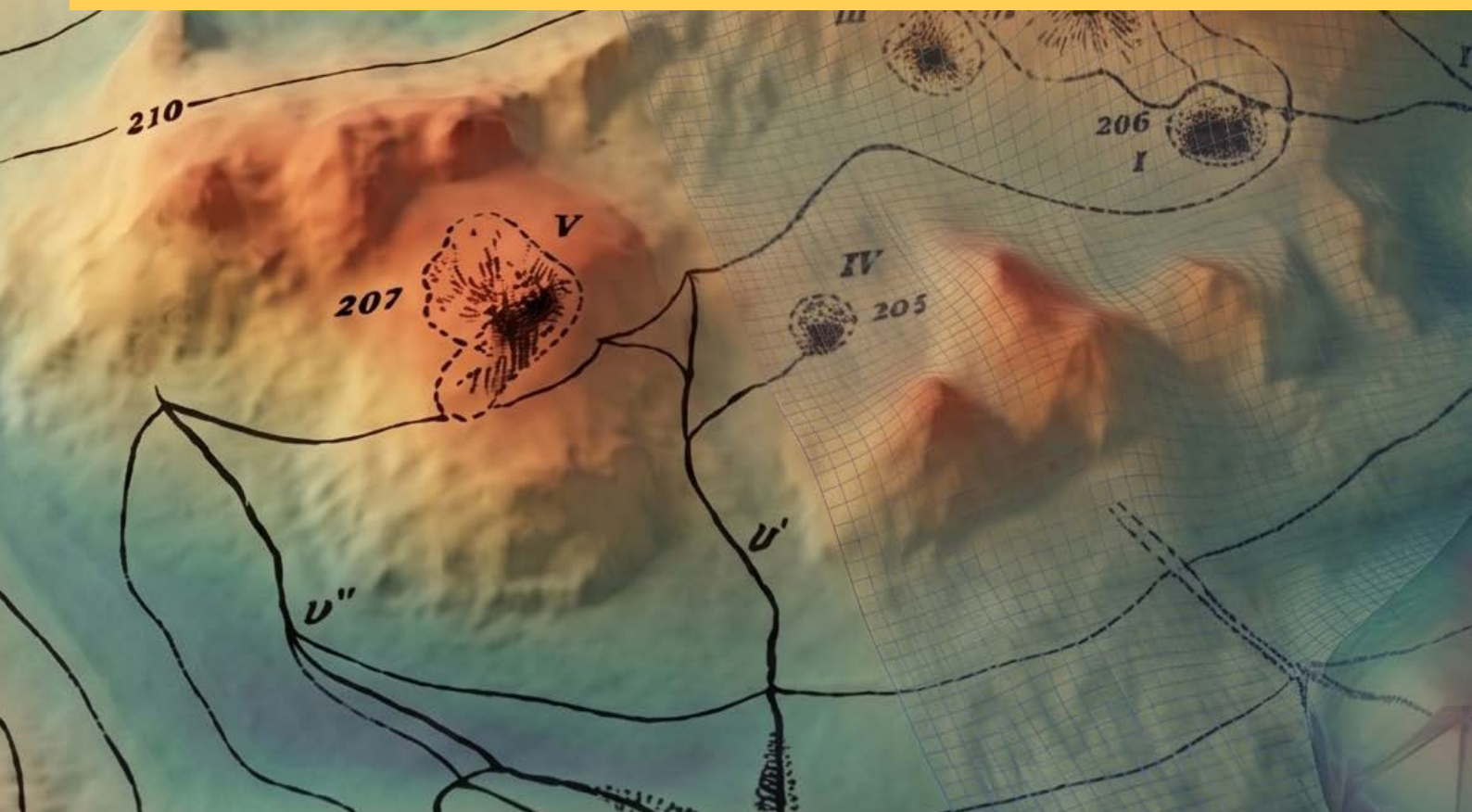




RAPPORTI TECNICI INGV

DigMap (MATLAB) - Digitalizzazione
e correzione di scala per mappe
topografiche storiche



ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

516

**Direttore Responsabile**

Daniela VERSACE

Editor in Chief

Milena MORETTI

Editorial Board

Laura ALFONSI

Sofia BARANELLO

Carla BARNABA

Christian BIGNAMI

Simona CARANNANTE

Maria Grazia CIACCIO

Luca COCCHI

Luigi CUCCI

Lorenzo CUGLIARI

Elisabetta D'ANASTASIO

Alessia DI CAPRIO

Roberto DI MARTINO

Domenico DI MAURO

Domenico DORONZO

Filippo GRECO

Alessandro IAROCCI

Mario MATTIA

Daniele MELINI

Anna NARDI

Lucia NARDONE

Marco OLIVIERI

Arianna PESCI

Pierangelo ROMANO

Paola SBARRA

Davide SCAFIDI

Maurizio SOLDANI

Sara STOPPONI

Umberto TAMMARO

David ZULIANI

REGISTRAZIONE AL TRIBUNALE DI ROMA N.173 | 2014, 23 LUGLIO

© INGV Istituto Nazionale di Geofisica e Vulcanologia

Rappresentante legale: Presidente INGV

Sede: Via di Vigna Murata, 605 | Roma

Tutti i diritti riservati

www.ingv.it



ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA

RAPPORTI TECNICI INGV

DigMap (MATLAB) - Digitalizzazione e
correzione di scala per mappe topografiche
storiche

*DigMap (MATLAB) - Digitization and scale
correction of historical topographic maps*

Arianna Pesci*, Daniele Randazzo

INGV | Istituto Nazionale di Geofisica e Vulcanologia, Sezione di Bologna

*Corresponding Author

Accettato 11 dicembre 2025 | Accepted 11 December 2025

Come citare | How to cite Pesci, A., Randazzo, D. (2026). DigMap (MATLAB) - Digitalizzazione e correzione di scala per mappe topografiche storiche. *Rapporti Tecnici INGV*, 516, 54 pp. <https://doi.org/10.13127/rpt/516>

In copertina Integrazione concettuale e modellazione 3D di dati cartografici storici e moderni | Cover Conceptual integration and 3D modeling of historical and modern cartographic data

516

INDICE

Riassunto	7
Abstract	7
Introduzione	7
1. Il software DigMap - Digitalizzatore di Mappe	8
2. Test e considerazioni	10
3. Esempio pratico: la mappa delle Salse di Nirano (1906)	11
3.1 Sequenza digitalizzazione	12
3.2 Sequenza mappatura	14
3.3 Nuvola di punti e modifiche successive	15
4. Conclusioni	16
Ringraziamenti	16
Disponibilità software	16
Bibliografia	16
Sitografia	17
Appendice 1	19
Appendice 2	33

Riassunto

In questo rapporto tecnico si presenta DigMap (Digitalizzatore di Mappe), un *software* sviluppato per rendere agevole la digitalizzazione di mappe storiche. Il programma consente di salvare i dati estratti (curve di livello o altri elementi) in formato testo, nella classica rappresentazione a tre colonne x, y, z. L'obiettivo principale è affrontare un problema frequente: l'assenza delle tavole originali con scala esplicita (1:N), che impedisce la conversione diretta tra pixel dell'immagine e misure reali. Il rapporto illustra le principali funzionalità del *software* attraverso esempi applicativi. Si tratta di strumenti semplici, pensati anche per utenti non esperti di informatica, che coniugano un approccio manuale con la chiarezza dei risultati. Le operazioni disponibili includono: caricamento di un'immagine, definizione della scala (provvisoria o definitiva), creazione interattiva di curve di livello, esportazione dei dati metrici, costruzione di un modello digitale del terreno (DEM), generazione di una nuvola di punti con densità a scelta dell'utente e, infine, importazione e riscalatura della nuvola stessa. Il *software*, al momento nato per la digitalizzazione delle mappe topografiche del 1906-1907 utilizzate nello studio delle salse (vulcani di fango) dell'Appennino Emiliano-Romagnolo, potrà in futuro essere ulteriormente automatizzato.

Abstract

This technical report presents DigMap (Digital Map Digitizer), a software developed to make the digitization of historical maps easier and more direct. The program allows saving the extracted data (contour lines or other elements) in text format, using the standard three-column structure x, y, z. The main goal is to address a common problem: the lack of original maps with an explicit scale (1:N), which prevents the direct conversion between image pixels and real-world measurements. The report describes the main functionalities of the software through application examples. The tools are intentionally simple, designed also for non-expert users, combining a manual approach with clear and reliable results. Available operations include: loading an image, defining a scale (temporary or final), interactive creation of contour lines, exporting metric data, building a digital elevation model (DEM), generating a point cloud with user-defined density, and finally importing and rescaling the cloud itself. The software was initially developed for the digitization of topographic maps produced in 1906–1907, used in a study of mud volcanoes (salse) in the Northern Apennines of Emilia-Romagna. In the future, further automation will be implemented.

Keywords | Mappe Storiche; Digitalizzazione; Scala | Historical Map; Digitalization; Scale

Introduzione

Negli ultimi decenni, molte tecniche di estrazione automatica da mappe digitalizzate si sono sviluppate soprattutto su prodotti cartografici recenti, caratterizzati da dati abbondanti, simbologie standardizzate e riferimenti geodetici affidabili, che permettono workflow GIS molto efficienti [Freeman and Pieroni, 1982; Chiang et al., 2009; Zatelli et al., 2019]. Allo stesso tempo, esiste un vasto insieme di mappe storiche, come quelle geologiche o geomorfologiche pubblicate in vecchie opere, che sono spesso disegnate a mano e presentano deformazioni planimetriche, scale variabili o informazioni metriche limitate. In questi casi, più che ottenere una precisione assoluta, l'interesse scientifico è riuscire a conservare e comprendere l'informazione

geometrica disponibile, anche se approssimativa, relativa a territori che oggi sono cambiati profondamente.

Storicamente, le misure altimetriche venivano realizzate con tecniche robuste (livellazione ottica o trigonometrica), rendendo la componente verticale generalmente attendibile. Al contrario, l'accuratezza planimetrica era più vulnerabile: misure con catene metriche o nastri potevano introdurre deviazioni cumulabili sul terreno, e l'orientamento dipendeva da strumenti sensibili a errori locali (come bussole e teodoliti semplici). Anche la fase di produzione cartacea e di successiva riproduzione (riduzioni, fotocopie, scansioni non controllate) introduceva ulteriori distorsioni, spesso non quantificabili a priori. Ne deriva che le mappe storiche, pur fondamentali per la ricostruzione di fenomeni territoriali e morfologici nel lungo periodo, richiedono approcci dedicati se si intende integrarle coerentemente con dati moderni.

In questo quadro si inserisce DigMap, che propone una soluzione operativa snella e controllata per la digitalizzazione di curve e linee rilevate in epoche passate. Il *software* consente principalmente di:

1. tracciare interattivamente gli elementi di interesse su mappe *raster* storiche;
2. assegnare valori altimetrici (anche di default, quando assenti);
3. esportare geometrie in formato metrico (XYZ);
4. correggere la scala planimetrica quando la riproduzione non coincide con quella nominale.

Tali esportazioni producono dati in chiave relativa, che solo in fasi successive di post-elaborazione saranno eventualmente allineati ai modelli recenti mediante algoritmi di *surface matching* [1] o altre soluzioni. Questo approccio consente di recuperare un'informazione imperfetta ma utilizzabile, mitigando le incertezze introdotte tanto dai rilievi originari quanto dalle successive riproduzioni e rendendo queste mappe confrontabili (seppur con le dovute cautele) con dati topografici, fotogrammetrici o geofisici contemporanei.

Sebbene esistano strumenti GIS avanzati per la gestione e l'analisi di dati geospaziali, quali QGIS [2], GRASS GIS [3] e le librerie GDAL/OGR [4], questi non affrontano in modo semplice e controllato l'estrazione interattiva di informazione geometrica da mappe storiche non georiferite, caratterizzate da deformazioni e incertezze sia metriche sia riproduttive, o comunque richiedono un livello di specializzazione e padronanza che implica tempi di apprendimento elevati.

Prima di procedere, è bene sottolineare che DigMap nasce da una necessità specifica: dare nuova luce a informazioni geografiche e morfologiche riportate in modo sommario all'interno di pubblicazioni geografiche e geologiche d'epoca. In tali documenti, le forme del terreno sono spesso rappresentate tramite serie di isolinee con quota indicata, prive di riferimenti planimetrici affidabili, di una scala certa o di un sistema di coordinate definito; la loro collocazione assoluta deriva unicamente dalla lettura delle relazioni testuali, di natura qualitativa e descrittiva. Per questa ragione, il carattere del lavoro è intrinsecamente relativo: l'obiettivo non è produrre un modello altimetrico autonomo e accurato in senso assoluto, bensì estrarre una geometria coerente delle curve, trasformandole in una nuvola di punti metrizzabile che potrà successivamente essere allineata e confrontata con dati e modelli recenti. In questo modo, mappe storiche apparentemente poco utilizzabili diventano una risorsa quantitativa per ricostruire l'evoluzione territoriale nel lungo periodo.

1. Il software DigMap - Digitalizzatore di Mappe

Lo *script* DigMap (abbreviazione di Digitalizzatore di Mappe) è stato sviluppato in ambiente MATLAB [5] per semplificare l'estrazione di informazioni quantitative da mappe storiche o da riproduzioni di tavole tecniche. La procedura è interattiva: l'operatore utilizza il *mouse* per ricalcare le linee di interesse sull'immagine, trasformandole in coordinate metriche.

Il funzionamento generale si articola nelle seguenti fasi principali:

1. caricamento dell'immagine: l'utente apre la mappa da elaborare. Una finestra iniziale permette di navigare nell'immagine e leggere le coordinate in *pixel*;
2. definizione della scala: l'immagine viene ricaricata e si selezionano due punti noti, definendo un segmento al quale si associa un valore metrico reale. Questo passaggio calibra l'immagine e trasforma i *pixel* in misure metriche;
3. estrazione delle curve di livello: l'operatore clicca sui profili da digitalizzare. Al termine del tracciamento (doppio *click*) viene richiesto di assegnare una quota; se non specificata, il valore predefinito è 0. Per ciascuna curva viene creato automaticamente un *file* TXT con la quota indicata nel nome;
4. visualizzazione 2D: i *file* delle linee estratte possono essere richiamati per la rappresentazione planimetrica. Ogni curva è visualizzata con un colore differente per facilitarne la distinzione;
5. creazione del modello digitale del terreno (DEM): dopo la selezione delle curve, DigMap propone automaticamente l'estensione spaziale dell'area e la griglia di calcolo, modificabili dall'operatore. Viene quindi generato un DEM e salvato in un file di testo contenente le coordinate (X, Y, Z) della superficie, pronto per l'esportazione in formati standard;
6. correzione della scala orizzontale: è possibile importare il *file* della nuvola e ridimensionarlo solo orizzontalmente. Questa funzione gestisce un problema frequente: le riproduzioni di mappe storiche spesso non conservano le proporzioni reali a causa di scansioni, fotografie o riduzioni/ingrandimenti. In questi casi la scala nominale (es. 1:1000) può non corrispondere a quella effettiva.

Lo schema di lavoro è riportato in Figura 1. Le curve di livello estratte diventano oggetti metrici, utilizzabili per costruire DEM e integrabili con altre fonti storiche o moderne.

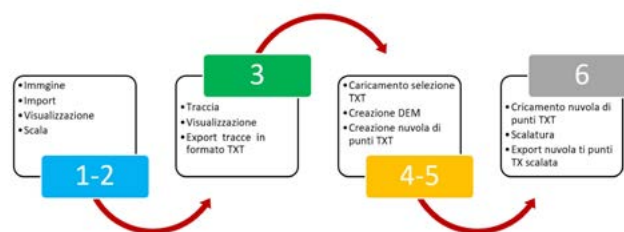


Figura 1 Diagramma del lavoro. Sequenza delle operazioni con riferimento ai punti descritti in elenco.

Figure 1 Workflow diagram. Sequence of operations with reference to the points described in the list.

Oltre alle caratteristiche sopra descritte, DigMap fornisce una serie di funzionalità operative che ne agevolano l'utilizzo in differenti contesti applicativi. L'interfaccia sviluppata in MATLAB guida l'utente in tutte le fasi della digitalizzazione, dall'apertura dell'immagine alla creazione del modello digitale, rendendo il *workflow* riproducibile anche da non esperti di programmazione. I *file* delle linee vengono salvati progressivamente, consentendo di interrompere e riprendere l'estrazione in qualunque momento e permettendo la selezione selettiva delle sole isolinee utili alle analisi successive.

L'approccio manuale, pur non automatizzato, garantisce un elevato controllo visivo sulla qualità della digitalizzazione, caratteristica particolarmente utile quando si lavora con mappe storiche che possono presentare irregolarità grafiche, deformazioni o linee poco leggibili. I dati generati (curve e DEM) sono esportabili in formati testuali *standard*, facilmente importabili in *software* di modellazione 3D o in pacchetti GIS, permettendo integrazioni con rilievi più recenti come dati LiDAR, fotogrammetria o modellazioni geofisiche.

Le funzionalità di correzione metrica costituiscono un elemento distintivo del *software*: la possibilità di intervenire successivamente sulla scala orizzontale della nuvola di punti consente di minimizzare le distorsioni introdotte dalle riproduzioni non uniformi delle mappe antiche, mantenendo coerente la componente verticale, storicamente più accurata grazie alla livellazione ottica.

Per una descrizione operativa dettagliata dell'interfaccia e delle singole procedure (selezione dei punti, assegnazione delle quote, impostazione della griglia e salvataggio dei risultati) si rimanda all'Appendice 1, nel quale vengono riportate le schermate principali e la sequenza completa delle azioni modulari previste dal *workflow*. Il codice completo, invece, nella sua versione più recente è riportato nell'Appendice 2.

In aggiunta al salvataggio dei singoli elementi digitalizzati, utile per mantenere separati oggetti concettualmente diversi ed eventualmente correggerli o riorganizzarli anche in fasi successive, è implementata una modalità opzionale di esportazione unificata.

Questa funzione consente di generare un unico *file* di progetto contenente l'insieme completo delle geometrie e delle relative quote, facilitando il ricaricamento del lavoro e il passaggio diretto alle fasi di elaborazione successive (es. ricostruzione della superficie o allineamento a modelli recenti).

La doppia modalità di salvataggio garantisce quindi flessibilità operativa: da un lato il controllo puntuale sulle singole curve, dall'altro la possibilità di disporre rapidamente della nuvola di punti complessiva per analisi o integrazioni esterne.

2. Test e considerazioni

Prima dell'applicazione su dati reali (dove verrà illustrata nel dettaglio la procedura), è stata condotta una verifica quantitativa dell'affidabilità del metodo, utilizzando superfici sintetiche prive di incertezze geometriche. In questo modo è stato possibile isolare e valutare i soli contributi di errore dovuti alla digitalizzazione manuale e all'interpolazione.

La principale fonte di incertezza nelle mappe storiche è rappresentata infatti dalla distorsione planimetrica, legata sia al rilievo originario sia ai passaggi di stampa e riproduzione. DigMap affronta il problema mediante una calibrazione interattiva della scala applicata direttamente sull'immagine, e consente di correggere selettivamente la componente orizzontale della nuvola di punti, mantenendo inalterate le quote verticali, di norma più affidabili. Per la costruzione del DEM viene adottata l'interpolazione "*natural*" di MATLAB, basata su triangolazione di Delaunay, che preserva le geometrie imposte dalle curve di livello ed evita oscillazioni non realistiche tipiche di *spline* o approcci che richiedono ipotesi aggiuntive (ad es. *kriging*).

Il test è stato realizzato mediante tre curve di livello sintetiche (0, 10 e 20 m), costituite da cerchi concentrici entro un'area di 100 m × 100 m. Le curve sono state rappresentate in un'immagine *raster* con caratteristiche analoghe a una tavola topografica (10 × 10 cm, spessore della linea ≈ 1 mm, scala ≈ 1:1000). Ciò implica che 1 mm sul disegno corrisponde a ~1 m nella realtà, costituendo già una soglia grafica di incertezza. Una digitalizzazione "speditiva" delle *isoipse* ha prodotto un campionamento limitato e volutamente non ottimizzato, in analogia al caso reale. Il confronto con i dati teorici ha mostrato un eccellente accordo: la deviazione *standard* delle differenze planimetriche è pari a circa 0.7 m, in linea con la precisione attesa dalla fonte grafica, con valori massimi entro ±2 m. Ne consegue che l'accuratezza ottenuta è dominata dalla qualità dell'immagine sorgente e non dal processo numerico di digitalizzazione e interpolazione.

Sebbene soluzioni GIS e librerie *open-source* (es. QGIS, GDAL) permettano di trattare modelli topografici complessi, esse richiedono dati già metrici e non integrano una fase di ricalco guidato né la correzione selettiva della sola scala planimetrica. DigMap si configura pertanto come strumento leggero e mirato alla valorizzazione quantitativa di mappe storiche non georiferite,

dove il controllo manuale delle geometrie rimane essenziale. Nel paragrafo seguente viene illustrato un caso studio reale (Salse di Nirano, MO), mentre i dettagli tecnici del codice MATLAB adottato sono riportati in Appendice 1.

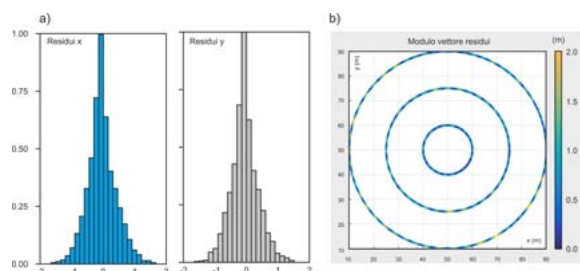


Figure 2 a) Istogramma dei residui planimetrici tra il modello teorico e i punti ottenuti dalla digitalizzazione (scala normalizzata), con evidenza dell'errore medio e della deviazione *standard*. b) Distribuzione spaziale dei residui sulle curve teoriche, rappresentata mediante codifica cromatica proporzionale alla differenza rispetto alle tracce digitalizzate: i margini entro ± 1 m risultano prevalenti, confermando un buon allineamento complessivo.

Figure 2 a) Histogram of planimetric residuals between the theoretical model and the points obtained by digitalization (normalized scale), highlighting mean error and standard deviation. b) Spatial distribution of residuals along the theoretical contours, using a color coding proportional to the differences with respect to the digitized traces. Most values fall within ± 1 m, confirming a generally good alignment.

3. Esempio pratico: la mappa delle Salse di Nirano (1906)

Nel cuore dell'Appennino Emiliano-Romagnolo, vicino a Sassuolo (MO), si trovano le Salse di Nirano, vulcani di fango costituiti da coni dai quali fuoriescono fango, acqua salata e gas, principalmente metano. La loro attività è dovuta a sedimenti argillosi profondi e alla risalita del gas sotterraneo. I coni, sebbene modesti in altezza, hanno un forte impatto visivo e scientifico: il fango emergente costruisce nuove formazioni, mentre le bolle di gas e acqua rendono la superficie instabile e in continua evoluzione. Già agli inizi del XX secolo, le salse erano oggetto di studi per comprendere la diagenesi dei sedimenti, la dinamica dei fluidi sotterranei e i rischi legati a fuoriuscite di gas e fango. Storicamente, esistono resoconti che descrivono fenomeni intensi e parossismi alle Salse di Nirano, talvolta illustrati anche con dipinti antichi; tuttavia, la documentazione tecnica più antica è costituita dai rilievi topografici e dalle immagini fotografiche di inizio Novecento. Nel sito della Regione Emilia-Romagna è possibile visionare le pubblicazioni che riportano informazioni storiche [6].

Un'opera molto interessante pubblicata nel 1907 come primo volume delle Memorie Geografiche a cura di Giotto Dainelli [1907] geografo e geologo di spicco, direttore della Rivista Geografica Italiana tra il 1907 e il 1919, riporta uno studio esteso e multidisciplinare delle salse dell'Appennino settentrionale.

Nella Figura 3 è mostrato un estratto del documento, contenente la mappa topografica dell'area [Biasutti, 1907] con una buona evidenza altimetrica, grazie a misure di livellazione, e una copertura planimetrica soddisfacente. L'immagine, scansionata dalla pubblicazione originale, è stata digitalizzata e resa disponibile dalla Biblioteca Nazionale Centrale di Roma [7] alla sezione "emeroteca". Come già accennato, la scala planimetrica non corrisponde esattamente alle dimensioni del foglio, quindi la mappa va elaborata e scalata per ottenere dati metrici corretti. DigMap è stato quindi utile per estrarre i dati e renderli utilizzabili per creare e ispezionare le mappe nella loro configurazione originale, risalente a oltre un secolo fa.

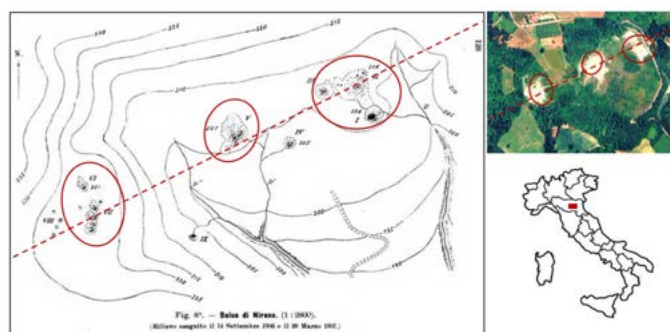


Figura 3 Rilievo topografico eseguito nel 1906 mediante livellazione. L'area corrisponde, in epoca passata, alla zona mostrata nel riquadro (Google Earth, 2025). I cerchi rossi indicano le principali bocche di emissione sulla mappa storica; la linea tratteggiata e i cerchi indicano i vulcanetti presenti oggi (2025) e il loro allineamento nello spazio (da Google Earth), riportati anche sulla mappa storica. L'assetto è evidentemente cambiato ma le variazioni non sono tema di questo lavoro.

Figure 3 Topographic survey carried out in 1906 using leveling. The area corresponds, at that time, to the zone shown in the inset (Google Earth, 2025). Red circles indicate the main emission vents on the historical map; the dashed line and circles show the small mud volcanoes present today (2025) and their spatial alignment (from Google Earth), also marked on the historical map. The layout has obviously changed, but these variations are not the focus of this work.

3.1 Sequenza digitalizzazione

Per iniziare l'analisi della mappa storica delle Salse di Nirano, l'utente apre DigMap in MATLAB e seleziona il file immagine da elaborare (Figura 4). Una volta caricata, la mappa può essere esplorata con il cursore per leggere le coordinate in pixel, utile per orientarsi e individuare i punti di interesse (Figura 5).

Figura 4 Apertura dell'interfaccia MATLAB di DigMap e selezione del file immagine contenente la mappa delle Salse di Nirano.

Figure 4 Opening the DigMap MATLAB interface and selecting the image file containing the map of the Salse di Nirano.

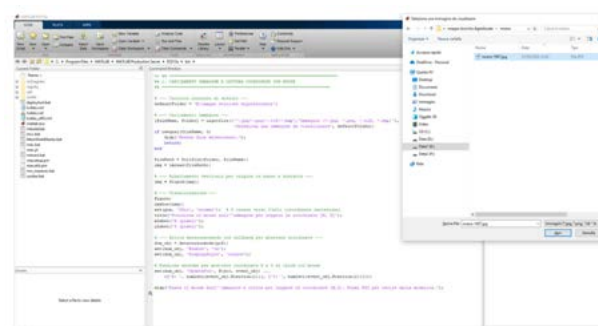
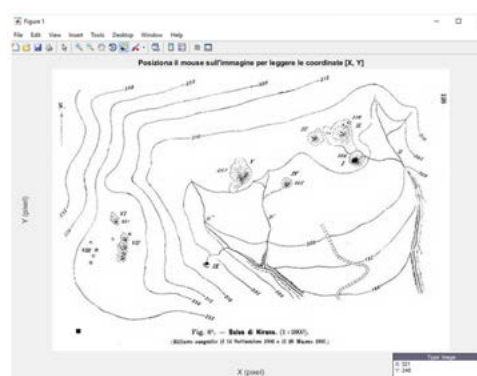


Figura 5 Visualizzazione della mappa caricata; l'utente può spostare il cursore per leggere le coordinate X e Y in pixel.

Figure 5 Display of the loaded map; the user can move the cursor to read X and Y coordinates in pixels.



Il passo successivo è la definizione della scala: l'utente seleziona due punti di riferimento noti e li collega con una linea, assegnando la distanza reale sul terreno. Questo consente di convertire correttamente le coordinate da *pixel* a metri (Figura 6).

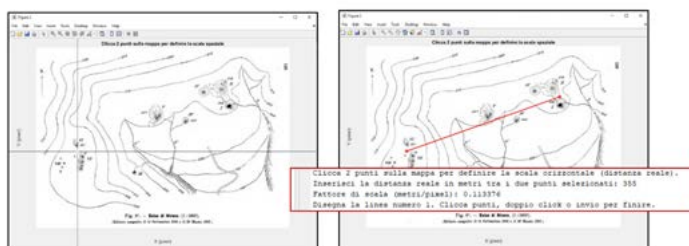


Figura 6 Definizione dei punti di riferimento per la scala: a) selezione dei due punti sulla mappa tramite linee incrociate; b) linea congiungente i punti; c) inserimento della lunghezza reale per calibrare il fattore di scala.

Figure 6 Definition of reference points for the scale: a) selecting two points on the map using intersecting lines; b) line connecting the points; c) entering the real distance to calibrate the scale factor.

Terminata la calibrazione, si procede alla digitalizzazione delle curve di livello. Cliccando lungo le linee di interesse, l'utente assegna a ciascuna quota un valore numerico; nell'esempio mostrato, la curva più bassa è stata impostata a 190 m (Figura 7).

```
Disegna la linea numero 1. Clicca punti, doppio click o invio per finire.
Inserisci quota per questa linea (default 0): 190
Linea salvata in E:\mappe storiche digitalizzate\nirano\linea_1_quota_190.txt
Vuoi tracciare ancora una linea? (1 = sì, 0 = no): 1
Disegna la linea numero 2. Clicca punti, doppio click o invio per finire.
```

Figura 7 Digitalizzazione interattiva di una linea di quota: l'utente clicca lungo la curva di livello e assegna la quota finale (190 m nel caso mostrato).

Figure 7 Interactive digitalization of a contour line: the user clicks along the line and assigns the final elevation (190 m in the example shown).

I file delle linee digitalizzate vengono quindi caricati nuovamente in DigMap, pronti per essere visualizzati e gestiti (Figura 8). Le isolinee importate possono essere rappresentate in scala reale, grazie all'applicazione del fattore di conversione dalle coordinate in *pixel* a quelle metriche (Figura 9).

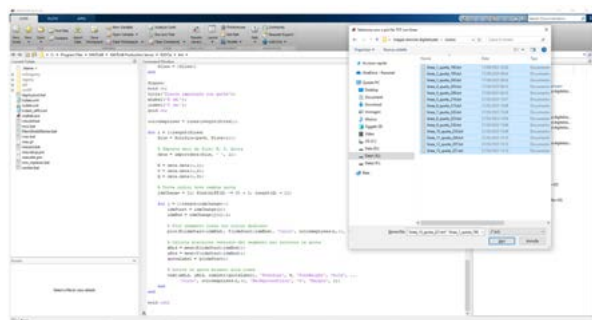
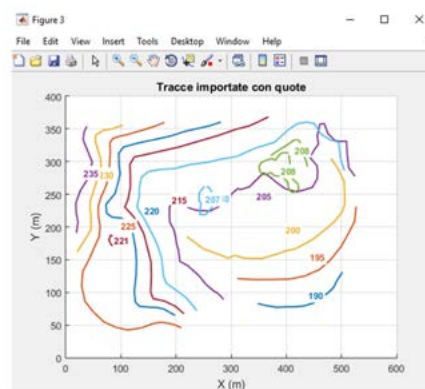


Figura 8 Interfaccia per il caricamento dei file delle linee digitalizzate salvati.

Figure 8 Interface for loading the saved digitalized line files.

Figura 9 Visualizzazione delle isolinee importate con le coordinate scalate in metri, mostrando la trasformazione da pixel a misura reale.

Figure 9 Display of imported contour lines with coordinates scaled to meters, showing the conversion from pixels to real measurements.



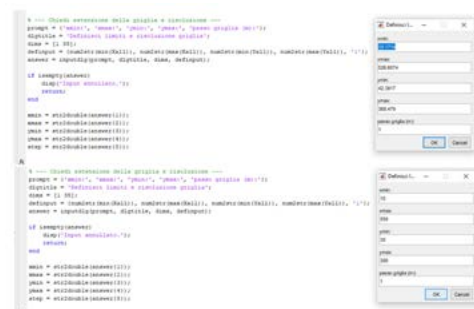
È anche implementata la possibilità di creare un file cumulativo con tutte le isolinee, in fase di salvataggio, per avere un riferimento più compatto e meglio gestibile laddove non sia di interesse la cernita di isolinee da utilizzare.

3.2 Sequenza mappatura

A questo punto si definiscono i limiti della griglia per costruire il modello digitale del terreno (DEM). Il software suggerisce valori di *default*, ma l'utente può modificarli a piacere e scegliere la risoluzione della griglia, impostata in questo esempio a 1 m, per ottenere un modello più dettagliato (Figura 10).

Figura 10 Definizione dei limiti della griglia per la costruzione del modello digitale del terreno (DEM): a) valori di *default* basati sui dati importati; b) valori ridefiniti dall'utente con passo griglia impostato a 1 m.

Figure 10 Definition of the grid limits for building the digital terrain model (DEM): (a) default values based on imported data; (b) user-defined values with grid step set to 1 m.



Con le tracce digitalizzate e i limiti della griglia definiti, DigMap genera il DEM. La figura mostra sia i punti originali delle curve sia la superficie interpolata, con le quote evidenziate tramite una scala colore compresa tra 190 e 235 m (Figura 11).

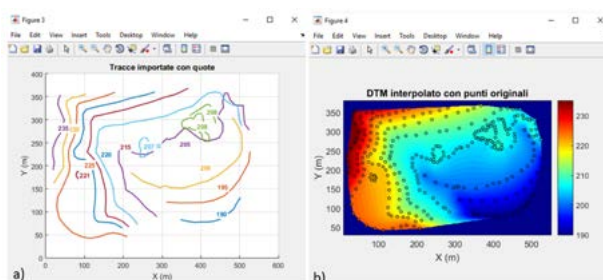


Figura 11 a) Coordinate delle tracce digitalizzate; b) corrispondente modello DEM con scala colore 190–235 m, sovrapposto alle coordinate originali.

Figure 11 a) Coordinates of the digitalized traces; b) corresponding DEM with color scale 190–235 m, overlaid on the original coordinates

3.3 Nuvola di punti e modifiche successive

Il modello può essere esportato come nuvola di punti in formato txt, rendendolo facilmente utilizzabile in altri *software* di modellazione o per analisi geofisiche (Figura 12).



Figura 12 Esportazione della nuvola di punti dal DEM: a) interfaccia DigMap per il salvataggio; b) nuovo file (nuvola.txt) salvato nella cartella contenente tutte le coordinate.
Figure 12 Exporting the point cloud from the DEM: a) DigMap interface for saving; b) TXT file containing all coordinates.

Se necessario, è possibile correggere la scala orizzontale della nuvola di punti, per compensare eventuali differenze tra la mappa digitale e la reale dimensione sul terreno. L'utente seleziona il file e imposta la percentuale di ridimensionamento desiderata (Figura 13).

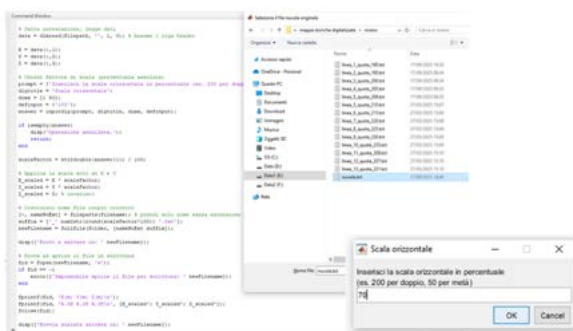


Figura 13 Caricamento della nuvola di punti per il nuovo ridimensionamento orizzontale; selezione della percentuale di scala tramite finestra di input.
Figure 13 Loading the point cloud for new horizontal scaling; selection of the scale percentage via input window.

Infine, la nuvola di punti scalata viene salvata con un nuovo nome che indica la modifica effettuata. Nel caso mostrato, il suffisso "_70" indica che le coordinate planimetriche sono state ridotte al 70% della scala precedente, mentre le quote verticali rimangono invariate (Figura 14).

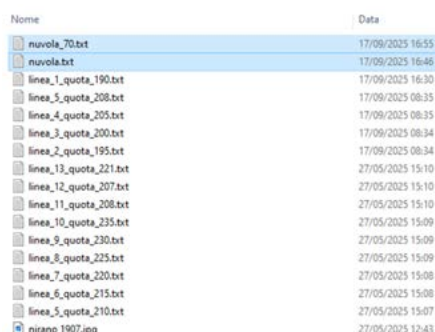


Figura 14 Salvataggio della nuvola scalata orizzontalmente; il suffisso (_70) indica l'applicazione del 70% della scala precedente alle componenti planimetriche.

Figure 14 Saving the horizontally scaled point cloud; the suffix (_70) indicates that 70% of the previous scale was applied to the planimetric components.

4. Conclusioni

DigMap si è rivelato uno strumento semplice ma funzionale, sviluppato per offrire una soluzione pratica al recupero di informazioni altimetriche da mappe storiche non georiferite e spesso affette da distorsioni.

La sua natura esplicitamente relativa lo rende adatto a dati originariamente concepiti per una rappresentazione qualitativa, curve di livello prive di riferimento planimetrico affidabile, trasformandoli in una risorsa utilizzabile in contesti di analisi attuali.

Il ricalco manuale e progressivo delle curve di livello consente un controllo completo del processo, permettendo di aggiornare o integrare selettivamente i dati senza dover ripetere operazioni già svolte. Le funzioni di esportazione permettono di ottenere rapidamente una nuvola di punti integrabile con modelli digitali del terreno più recenti, ampliando il valore informativo di materiali storici altrimenti statici.

Un ulteriore elemento di versatilità è dato dalla possibilità di modificare a posteriori la scala orizzontale della nuvola di punti, così da compensare le inevitabili incertezze planimetriche mantenendo invariata la quota, generalmente più affidabile, in quanto spesso derivata da livellazioni storiche.

Dal punto di vista implementativo, DigMap è strutturato come una cascata ordinata di blocchi operativi, ciascuno dedicato a una specifica funzione. Tale architettura non solo ne semplifica l'utilizzo per utenti non esperti di programmazione, ma favorisce anche future espansioni: funzioni, metodi e procedure possono essere sostituiti, aggiornati o potenziati in modo chiaro e pulito, mantenendo leggibilità e stabilità del codice.

Attualmente, DigMap è adottato dalla WP2 della Sezione di Bologna dell'INGV nell'ambito del progetto Pianeta Dinamico ProMud [8], a supporto dello studio dei vulcani di fango mediante la ricostruzione tridimensionale della loro evoluzione morfologica. Sono già previsti sviluppi per ridurre ulteriormente l'intervento manuale, automatizzando parte della digitalizzazione e migliorando l'allineamento ai modelli recenti. In ogni caso, DigMap manterrà la sua identità originaria: uno strumento leggero, modulare e mirato, pensato per restituire nuova utilità a dati storici altrimenti confinati nella sola dimensione illustrativa.

Ringraziamenti

Gli autori ringraziano Cecilia Ciatti, del Servizio Rari della Biblioteca Umanistica dell'Università di Firenze, per aver fornito supporto nell'individuazione e nell'accesso online alla pubblicazione della Rivista Geografica Italiana. Grazie a Paola Cusano, coordinatrice del progetto ProMud.

Disponibilità software

Il *software* DigMap, nella sua versione più recente, è in Appendice 2 di questo Rapporto Tecnico e gli eventuali aggiornamenti o modifiche successive saranno resi disponibili dagli autori su richiesta, nel rispetto delle condizioni previste dalla collana Rapporti Tecnici INGV.

Bibliografia

- Biasutti, R. (1907). Le salse dell'Appennino settentrionale. In: *Memorie Geografiche*, 1, *Supplemento alla Rivista Geografica Italiana*. Tipografia M. Ricci, Firenze.
- Chiang, Y.-Y., Knoblock, C.A., Shahabi, C., Chen, C.-C. (2009). Automatic and accurate

extraction of road intersections from raster maps. *Geoinformatica*, 13, 121–157. <https://doi.org/10.1007/s10707-008-0046-3>

Dainelli, G. (1907). *Memorie Geografiche*, 1. *Supplemento alla Rivista Geografica Italiana*. Tipografia M. Ricci, Firenze.

Freeman, H., and Pieroni, E. (1982). *Map data processing*. Academic Press.

Zatelli, P., Gobbi, S., Tattoni, C., La Porta, N., Ciolli, M. (2019). Object-based image analysis for historic maps classification. *ISPRS Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*. <https://doi.org/10.5194/isprs-archives-XLII-4-W14-247-2019>

Sitografia

[1] <https://www.polyworks.com>

[2] <https://qgis.org>

[3] <https://gdal.org>

[4] <https://grass.osgeo.org>

[5] <https://it.mathworks.com>

[6] <https://ambiente.regione.emilia-romagna.it>

[7] <http://digitale.bnc.roma.sbn.it>

[8] <https://progetti.ingv.it/it/promud>

APPENDICE 1

Note operative DigMap

Qui si forniscono informazioni di dettaglio al fine di utilizzare al meglio il software ma anche poterlo cambiare facilmente o aggiornare in base alle diverse esigenze.

Caricamento immagine e lettura

Questo blocco di codice serve a caricare una mappa storica digitalizzata e a permettere all'utente di leggere le coordinate dei punti cliccati direttamente sull'immagine. È la fase preliminare di DigMap, utile per familiarizzare con l'immagine e avere un'idea dei pixel corrispondenti alle posizioni di interesse.

Il funzionamento passo per passo vede le stringhe di comando nella sequenza necessaria:

1) Cartella iniziale

```
defaultFolder = 'E:\mappe storiche digitalizzate';
```

Qui viene impostata la cartella in cui il programma cercherà inizialmente le immagini. In questo modo non bisogna navigare ogni volta tra le cartelle del computer.

2) Selezione del file

```
[fileName, folder] = uigetfile({'*.jpg;*.png;*.tif;*.bmp', ...}, 'Seleziona una immagine da visualizzare', defaultFolder);
```

Viene aperta una finestra di dialogo che permette di scegliere il file immagine. Si possono selezionare JPG, PNG, TIF o BMP. Se l'utente preme "Annulla", il programma interrompe l'esecuzione.

3) Lettura dell'immagine

```
filePath = fullfile(folder, fileName);  
img = imread(filePath);
```

Qui l'immagine scelta viene caricata in memoria come matrice di pixel.

4) Ribaltamento verticale

```
img = flipud(img);
```

Le immagini digitali hanno l'origine in alto a sinistra (pixel 1,1). Ribaltandola verticalmente, l'origine viene spostata in basso a sinistra, come nelle coordinate cartesiane tradizionali, rendendo più naturale lavorare con mappe topografiche.

5) Visualizzazione dell'immagine

```
figure;  
imshow(img);  
set(gca, 'YDir', 'normal');  
title('Posiziona il mouse sull'immagine per leggere le coordinate [X, Y]');  
xlabel('X (pixel)');  
ylabel('Y (pixel)');
```

Viene aperta una finestra grafica con l'immagine. "set(gca, 'YDir', 'normal')" garantisce che l'asse Y cresca verso l'alto, come in un sistema cartesiano. Il titolo e le etichette guidano l'utente su cosa fare.

6) Attivazione cursore

```
dcm_obj = datacursormode(gcf);  
set(dcm_obj, 'Enable', 'on');  
set(dcm_obj, 'DisplayStyle', 'window');
```

Il `datacursormode` permette di cliccare sull'immagine e leggere le coordinate X e Y del pixel. Lo stile "window" fa apparire le coordinate in una piccola finestra *pop-up*.

7) Funzione anonima `datacursor`

```
set(dcm_obj, 'UpdateFcn', @(src, event_obj) ...  
{'X: ', num2str(event_obj.Position(1))}, ['Y: ', num2str(event_obj.Position(2))]);
```

Questa riga definisce come vengono visualizzate le coordinate: quando l'utente clicca, appare una finestra con X e Y.

8) Messaggio

```
disp('Passa il mouse sull'immagine e clicca per leggere le coordinate (X,Y). Premi ESC per uscire dalla  
modalità.');
```

Messaggio che guida l'utente: si può cliccare per leggere i valori dei pixel e uscire premendo ESC.

Notare che: questo *step* non modifica la mappa, serve solo a visualizzare e leggere i punti. La precisione è limitata ai pixel dell'immagine e serve per esplorare la mappa e avere una prima stima delle posizioni prima di assegnare quote e costruire il modello 3D. Inoltre la ripetuta importazione (vedi dopo) permette anche di avere in parallelo due figure. Una su cui si opera e una di riferimento.

Definizione di scala e disegno

Questo blocco di codice ha due scopi principali: stabilire la scala reale della mappa e permettere all'utente di tracciare le linee di interesse con la quota associata, salvando i dati in file di testo. Come nel primo blocco, si inizia selezionando un'immagine dalla cartella di default. L'immagine viene letta e ribaltata verticalmente con "flipud" in modo che l'origine sia in basso a sinistra, coerente con la rappresentazione cartesiana. Viene quindi visualizzata con "imshow", impostando gli assi con Y crescente verso l'alto.

La definizione della scala serve a convertire le distanze misurate in pixel in metri reali sul terreno. L'utente clicca due punti sulla mappa che rappresentano una distanza nota. Il codice calcola la distanza in pixel (`dPixel`) tra i due punti e richiede di inserire la distanza reale corrispondente (`dReal`). Il fattore di scala viene quindi calcolato come: $\text{scaleFactor} = \text{dReal} / \text{dPixel}$.

Questo passaggio è cruciale perché permette di trasformare tutte le linee tracciate successivamente da pixel a metri reali.

Per il disegno delle linee e l'assegnazione delle quote, il programma entra in un ciclo "while" che consente di tracciare più linee. Per ogni linea, l'utente clicca lungo il percorso desiderato. Dopo il doppio click o la conferma, viene assegnata una quota (se non specificata, il valore predefinito è 0). Le coordinate X e Y vengono scalate usando `scaleFactor` e combinate con la quota costante in un vettore. I dati vengono salvati in un file TXT separato, con il nome che include il numero della linea e la quota. Ogni linea è salvata autonomamente, permettendo di selezionarle o combinarle per ulteriori elaborazioni.

Al termine di ogni linea, il programma chiede all'utente se desidera tracciare un'altra linea; in caso contrario, il ciclo termina. La descrizione precedente è elencata a punti con riferimento al codice:

1) Cartella

```
defaultFolder = 'E:\mappe storiche digitalizzate';
```

Qui si imposta nuovamente la cartella di default dove cercare le immagini. Serve solo per comodità, così non bisogna navigare ogni volta.

2) Finestra

```
[fileName, folder] = uigetfile({'*.jpg;*.png;*.tif;*.bmp','Immagini (*.jpg, *.png, *.tif, *.bmp)'), ...
```

```
'Seleziona una immagine da visualizzare', defaultFolder);
```

Questa riga apre una finestra di dialogo che permette di selezionare l'immagine della mappa. Solo i formati immagine comuni sono visibili.

3) Lettura immagine

```
img = imread(filePath);  
img = flipud(img);
```

Si legge l'immagine e si ribalta verticalmente ("flipud") per far sì che l'origine (0,0) sia in basso a sinistra, come nei sistemi cartesiani standard.

4) Visualizzazione

```
figure;  
imshow(img);  
set(gca, 'YDir', 'normal');  
title('Clicca 2 punti sulla mappa per definire la scala spaziale');
```

Si visualizza la mappa e si imposta l'asse Y che cresce verso l'alto. Qui l'utente deve cliccare due punti per definire la scala reale.

5) Definizione due punti

```
[xScale, yScale] = ginput(2);
```

"ginput(2)" permette di catturare le coordinate di due punti cliccati dall'utente sulla mappa. Serviranno per calcolare la scala tra pixel e metri reali.

6) Calcolo distanza punti

```
dPixel = sqrt( (xScale(2) - xScale(1))^2 + (yScale(2) - yScale(1))^2 );  
dReal = input('Inserisci la distanza reale in metri tra i due punti selezionati: ');  
scaleFactor = dReal / dPixel;
```

Qui viene calcolata la distanza tra i due punti in pixel (dPixel) e si chiede la distanza reale sul terreno (dReal). Il fattore di scala trasforma i pixel in metri.

7) Linea congiunzione

```
[x, y] = getline(gca, 'closed', false);
```

Questa funzione permette di tracciare una linea a mano libera sulla mappa cliccando punti successivi. closed=false significa che la linea non deve chiudersi automaticamente.

8) Tracciatura linee

```
quota = input('Inserisci quota per questa linea (default 0): ');  
x_scaled = x * scaleFactor;  
y_scaled = y * scaleFactor;  
dati = [x_scaled, y_scaled, quotaVec];
```

Dopo aver tracciato la linea, si assegna una quota. Poi le coordinate vengono convertite in metri usando il fattore di scala e combinate con la quota in un array pronto da salvare.

9) Salvataggio coordinate linee

```
nomeFile = fullfile(filepath, sprintf('linea_%d_quota_%g.txt', lineCount, quota));  
fid = fopen(nomeFile, 'w');  
fprintf(fid, 'X(m) Y(m) Quota\n');  
fprintf(fid, '%f %f %f\n', dati);  
fclose(fid);
```

Infine, la linea viene salvata in un file .txt con intestazione e dati. Il nome del file contiene il

numero della linea e la quota, così ogni linea è facilmente identificabile.

10)

```
rispostaNum = input('Vuoi tracciare ancora una linea? (1 = sì, 0 = no): ');
```

Alla fine, il programma chiede se si vogliono tracciare altre linee. Se l'utente risponde no, il ciclo termina.

In breve, questo blocco di codice consente di trasformare una mappa digitale in dati reali in metri, linea per linea, assegnando anche quote e salvando tutto in file pronti per ulteriori elaborazioni.

Oltre al salvataggio di ciascuna linea in un file dedicato, è stata introdotta un'opzione per esportare tutte le linee insieme in un unico file di testo.

Questa modalità permette una gestione più semplice del progetto e un uso più immediato dei dati nelle fasi successive (interpolazione e generazione della nuvola di punti).

11) Salvataggio unico di tutte le linee digitalizzate

Oltre al salvataggio di ogni linea in un file .txt separato, è possibile esportare l'intero dataset in un unico file, utile per la successiva interpolazione del modello digitale. Ogni volta che una linea viene tracciata, i suoi punti vengono memorizzati in una variabile cumulativa (allData). Al termine del processo, l'utente può decidere di esportare questo dataset aggregato.

```
% Inizializzazione del vettore cumulativo
allData = [];
% Ciclo principale di digitalizzazione linee
lineCount = 1;
rispostaNum = 1;
while rispostaNum == 1
    % Tracciamento linea
    [x, y] = getline(gca, 'closed', false);
    quota = input('Inserisci quota per questa linea (default 0): ');
    if isempty(quota), quota = 0; end
    % Scalatura
    x_scaled = x * scaleFactor;
    y_scaled = y * scaleFactor;
    quotaVec = quota * ones(size(x_scaled));
    dati = [x_scaled, y_scaled, quotaVec];
    % Accumulo in unico array
    allData = [allData; dati];
    % Salvataggio singolo file linea
    nomeLineFile = fullfile(filepath, sprintf('linea_%d_quota_%g.txt', lineCount, quota));
    fid = fopen(nomeLineFile, 'w');
    fprintf(fid, 'X(m) Y(m) Quota\n');
    fprintf(fid, '%f %f %f\n', dati);
    fclose(fid);
    lineCount = lineCount + 1;
    rispostaNum = input('Vuoi tracciare ancora una linea? (1 = sì, 0 = no): ');
end

% Salvataggio finale in unico file
salvaUnico = input('Vuoi salvare tutte le linee in un unico file? (1 = sì, 0 = no): ');
if salvaUnico == 1
```

```

unicoFile = fullfile(filepath, 'linee_tutte.txt');
fid = fopen(unicoFile, 'w');
fprintf(fid, 'X(m) Y(m) Z(m)\n');
fprintf(fid, '%f %f %f\n', allData');
fclose(fid);
disp(['File completo salvato in: ' unicoFile]);
end

```

Grafico linee e quote

Questo pezzo di DigMap serve a importare le linee precedentemente digitalizzate e salvate in file .txt, associando a ciascuna il valore della quota. Il codice poi plotta tutte le linee in un grafico 2D, usando colori distinti per ogni file e scrivendo il valore della quota lungo ogni segmento. In questo modo si ottiene una rappresentazione visiva immediata e chiara dei dati estratti.

In elenco i punti con riferimento al codice:

1) Selezione file da caricare

```

defaultPath = 'E:\mappe storiche digitalizzate';
[files, path] = uigetfile(fullfile(defaultPath, '*.txt'), ...
    'Seleziona uno o più file TXT con linee', 'MultiSelect', 'on');
if isequal(files, 0)
    disp('Nessun file selezionato.');
```

```

return;
end

```

Qui si apre una finestra di dialogo per scegliere uno o più file .txt contenenti le linee digitalizzate con le quote.

Se non si seleziona nulla, il programma esce.

2) Gestione del singolo file

```

if ischar(files)
    files = {files};
end

```

Se l'utente seleziona un solo file, MATLAB lo interpreta come stringa. Questo blocco lo converte in cell array, così il ciclo successivo funziona sempre.

3) Preparazione grafico

```

figure;
hold on;
title('Tracce importate con quote');
xlabel('X (m)');
ylabel('Y (m)');
grid on;
colormapLines = lines(length(files));

```

Crea una figura e imposta il grafico con titolo, etichette e griglia.

colormapLines assegna colori distinti per ogni file per rendere visivamente chiara la distinzione tra le linee.

4) Ciclo importazione e plot

```

for i = 1:length(files)
    file = fullfile(path, files{i});
    % Importa dati da file: X, Y, Quota

```

```
data = importdata(file, ',', 1);
X = data.data(:,1);
Y = data.data(:,2);
Q = data.data(:,3);
```

Ogni file viene caricato in MATLAB usando importdata. Si estraggono le colonne X, Y e la quota Q.

5) Individuazione cambio quota

```
idxChange = [1; find(diff(Q) ~= 0) + 1; length(Q) + 1];
diff(Q) calcola le differenze successive tra le quote.
```

Quando cambia il valore della quota, viene salvato l'indice per separare i segmenti omogenei (stessa quota).

6) Plot linee e quota

```
for j = 1:length(idxChange)-1
    idxStart = idxChange(j);
    idxEnd = idxChange(j+1)-1;
    plot(X(idxStart:idxEnd), Y(idxStart:idxEnd), 'Color', colormapLines(i,:), 'LineWidth',
1.5);
    xMid = mean(X(idxStart:idxEnd));
    yMid = mean(Y(idxStart:idxEnd));
    quotaLabel = Q(idxStart);
    text(xMid, yMid, num2str(quotaLabel), 'FontSize', 9, 'FontWeight', 'bold', ...
'Color', colormapLines(i,:), 'BackgroundColor', 'w', 'Margin', 1);
end
```

Ogni segmento di linea con quota costante viene plottato con il colore assegnato. xMid e yMid individuano il centro del segmento per scrivere il valore della quota accanto alla linea.

La funzione text permette di visualizzare chiaramente il valore della quota sul grafico, con sfondo bianco per renderlo leggibile.

7) Chiusura del grafico

```
hold off;
```

Termina la modalità "hold" del grafico. Tutti i segmenti sono ora visualizzati insieme, ciascuno con colore e quota. Questo step è essenziale per verificare visivamente la correttezza della digitalizzazione prima di passare alla generazione del DEM. L'uso di colori distinti e dei testi per le quote rende facile identificare eventuali errori o anomalie nei dati.

Import dei file digitalizzati

Questa sezione di DigMap serve a importare i file con le linee digitalizzate e generare una griglia per l'interpolazione, impostando così i limiti spaziali e la risoluzione del modello digitale del terreno (DEM). In pratica, qui si raccolgono tutti i punti XYZ e si definisce l'area di interesse e la densità della griglia su cui poi si interpoleranno i dati. Segue la spiegazione *step by step*:

1) Selezione multipla dei file

```
[files, folder] = uigetfile('E:\mappe storiche digitalizzate\*.txt', ...
'Seleziona i file di quota (x,y,z)', 'MultiSelect', 'on');
if isequal(files, 0)
    disp('Nessun file selezionato.');
```

```
return;
end
```

Viene aperta una finestra di dialogo per scegliere uno o più file .txt contenenti le linee con quote.

Se non si seleziona nulla, il programma termina.

2) Gestione del singolo file

```
if ischar(files)
    files = {files};
end
```

Se viene selezionato un solo file, MATLAB lo interpreta come stringa. Questo blocco lo converte in *cell array*, così i cicli successivi funzionano senza problemi.

3) Inizializzazione dei vettori

```
Xall = [];
Yall = [];
Zall = [];
```

Si creano vettori vuoti che accumuleranno tutte le coordinate dei file selezionati.

4) Importazione dati dai file

```
for i = 1:length(files)
    filePath = fullfile(folder, files{i});
    data = dlmread(filePath, ',', 1, 0); % skip header 1 riga
    Xall = [Xall; data(:,1)];
    Yall = [Yall; data(:,2)];
    Zall = [Zall; data(:,3)];
end
```

“*dlmread*” legge i dati saltando la prima riga di intestazione (‘X Y Quota’).

Tutti i punti dei vari file vengono concatenati nei vettori Xall, Yall, Zall.

5) Limiti e passo griglia

```
prompt = {'xmin:', 'xmax:', 'ymin:', 'ymax:', 'passo griglia (m):'};
dlgtitle = 'Definisci limiti e risoluzione griglia';
dims = [1 35];
definput = {num2str(min(Xall)), num2str(max(Xall)), num2str(min(Yall)), num2str(max(Yall)), '1'};
answer = inputdlg(prompt, dlgtitle, dims, definput);
if isempty(answer)
    disp('Input annullato.');
```

```
    return;
end
```

Si apre una finestra di dialogo per definire xmin, xmax, ymin, ymax e il passo della griglia (risoluzione).

Come default, il programma propone i valori minimi e massimi dei dati importati e un passo di 1 metro.

6) Salvataggio dati

```
xmin = str2double(answer{1});
xmax = str2double(answer{2});
ymin = str2double(answer{3});
ymax = str2double(answer{4});
step = str2double(answer{5});
```

Le stringhe inserite dall'utente vengono convertite in numeri e salvate nelle variabili corrispondenti.

Questo *step* è cruciale perché definisce l'estensione spaziale e la densità della griglia su cui poi

si calcolerà l'interpolazione del DEM. L'utente può adattare i limiti e la risoluzione, scegliendo tra un modello più dettagliato (passo piccolo) o uno più veloce da calcolare (passo grande).

Interpolazione

In questo passaggio, DigMap prende le linee importate e le loro quote e crea una griglia regolare su cui viene interpolato un modello digitale (DEM). Questo permette di ottenere una rappresentazione continua dell'altimetria, visualizzabile in 2D con colori e punti originali. L'utente può anche personalizzare la scala colori per evidenziare meglio le differenze di quota.

1) Creazione griglia regolare

```
xq = xmin:step:xmax;  
yq = ymin:step:ymax;  
[Xq, Yq] = meshgrid(xq, yq);
```

xmin, xmax, ymin, ymax e step sono stati definiti dall'utente in precedenza.

"meshgrid" costruisce una griglia 2D di coordinate X e Y dove l'interpolazione sarà effettuata. Ogni punto (Xq(i,j), Yq(i,j)) corrisponde a una posizione nello spazio su cui si stima la quota.

2) Interpolazione

```
Zq = griddata(Xall, Yall, Zall, Xq, Yq, 'natural');
```

Qui viene calcolata la quota Zq su ogni punto della griglia usando i dati originali (Xall, Yall, Zall). Il metodo 'natural' effettua una interpolazione naturale basata sui triangoli di Delaunay, rispettando bene la geometria dei dati circostanti. In sostanza, stima le quote tra le linee digitalizzate per creare una superficie continua.

3) Scala colore

```
promptC = {'Valore minimo scala colori:', 'Valore massimo scala colori:'};  
dlgtitleC = 'Imposta scala colori';  
definputC = {num2str(nanmin(Zq(:))), num2str(nanmax(Zq(:)))};  
answerC = inputdlg(promptC, dlgtitleC, dims, definputC);
```

L'utente può decidere i valori minimo e massimo della colorbar per evidenziare particolari range di quota.

Se l'utente annulla la finestra "(isempty(answerC))", il codice interrompe la procedura.

4) Visualizzazione 2D

```
figure;  
imagesc(xq, yq, Zq);  
set(gca, 'YDir', 'normal');  
colormap(jet);  
caxis([cmin cmax]);  
colorbar;  
title('DEM interpolato con punti originali');  
xlabel('X (m)');  
ylabel('Y (m)');  
axis equal tight;  
hold on;
```

"imagesc" visualizza le quote interpolati con colori sulla griglia. "colormap(jet)" applica una mappa colori predefinita (da blu a rosso). "caxis([cmin cmax])" impone la scala colori scelta dall'utente. "hold on" prepara la figura per plottare altri elementi, come i punti originali.

5) Plot dei punti originali

```
scatter(Xall, Yall, 15, Zall, 'filled', 'MarkerEdgeColor', 'k');
```

Sovrappone i punti digitalizzati originari sulle linee interpolate.

Il colore dei punti corrisponde alla quota, mentre il bordo nero (MarkerEdgeColor) li rende più visibili.

6) Finitura

```
set(gca, 'FontSize', 12);  
hold off;
```

Imposta una dimensione leggibile per le etichette degli assi. “hold off” conclude il *plotting* per questa figura.

Questa sezione trasforma un insieme sparso di linee con quote in un modello digitale continuo, pronto per analisi topografiche, visualizzazioni 3D o esportazione come nuvola di punti. È fondamentale perché permette di avere un DEM completo anche se le linee originali erano poche o irregolari.

Salvataggio nuvola di punti

In questa fase, il modello digitale creato (DEM) viene trasformato in una nuvola di punti ordinata, pronta per essere salvata in un file txt standard. Questo permette di esportare le coordinate X, Y, Z in un formato utilizzabile per altre analisi o software di modellazione 3D.

Step dettagliati e spiegazioni:

1) Trasformazione delle matrici in vettori colonna

```
Xv = Xq(:);  
Yv = Yq(:);  
Zv = Zq(:);
```

Le matrici Xq, Yq, Zq rappresentano la griglia regolare del DEM.

Con “(:)” vengono trasformate in vettori colonna, un passo necessario per salvare i dati in formato tabellare.

2) Selezione dei punti validi

```
validIdx = ~isnan(Zv);  
Xv_valid = Xv(validIdx);  
Yv_valid = Yv(validIdx);  
Zv_valid = Zv(validIdx);
```

Alcuni punti della griglia potrebbero non avere dati (NaN), ad esempio perché fuori dall’area coperta dalle linee originali.

Il codice filtra questi punti, conservando solo quelli con valori validi di quota.

3) Preparazione del file di output

```
outputFile = fullfile(folder, 'nuvola.txt');  
fid = fopen(outputFile, 'w');  
if fid == -1  
    error('Impossibile aprire il file per la scrittura.');
```

end

Il file verrà salvato nella stessa cartella dell’immagine originale.

Se il file non può essere creato o aperto, il codice mostra un messaggio di errore.

4) Scrittura dei dati nel file

```
fprintf(fid, 'X(m) Y(m) Z(m)\n');  
fprintf(fid, '%.3f %.3f %.3f\n', [Xv_valid; Yv_valid; Zv_valid]);
```

La prima riga è un'intestazione, utile per identificare le colonne.
Le coordinate vengono scritte con tre decimali, in metri, seguendo lo standard X Y Z.

5) Chiusura del file e conferma

```
fclose(fid);  
disp(['Nuvola di punti salvata in: ' outputFile]);
```

“fclose” chiude il file, rendendo sicuro il salvataggio.

Il messaggio finale conferma all'utente dove è stata salvata la nuvola di punti.

Questa nuvola di punti è ora pronta per essere importata in software GIS, CAD o per ulteriori elaborazioni geofisiche o fotogrammetriche. È il formato più versatile per integrare dati storici con rilievi moderni, modellazioni 3D o analisi quantitative.

Riassegnazione scala (opzionale)

Questa parte del software permette di correggere la scala orizzontale di una nuvola di punti già esistente, operazione utile quando la mappa originale o la sua riproduzione digitale non rispettano perfettamente la scala dichiarata. La quota verticale rimane invariata, perché è generalmente più affidabile nelle mappe storiche. *Step* dettagliati e spiegazioni:

1) Selezione del file della nuvola originale

```
[filename, folder] = uigetfile(...);  
if isequal(filename,0)  
    disp('Nessun file selezionato.');    return;  
end  
filepath = fullfile(folder, filename);
```

L'utente seleziona il file .txt contenente le coordinate XYZ della nuvola. Se nessun file viene selezionato, il programma termina senza errori.

2) Lettura dei dati e separazione in colonne

```
data = dlmread(filepath, ',', 1, 0);  
X = data(:,1);  
Y = data(:,2);  
Z = data(:,3);
```

La funzione “dlmread” legge il file saltando la prima riga (intestazione). Le colonne corrispondono a coordinate X, Y e Z.

3) Richiesta del fattore di scala

```
prompt = {'Inserisci la scala orizzontale in percentuale ...'};  
answer = inputdlg(prompt, dlgtitle, dims, definput);  
scaleFactor = str2double(answer{1}) / 100;
```

L'utente inserisce il valore della scala orizzontale come percentuale (es. 200 = doppio, 50 = metà). Il fattore viene convertito in formato numerico frazionario (1 = 100%).

4) Applicazione della scala

```
X_scaled = X * scaleFactor;  
Y_scaled = Y * scaleFactor;  
Z_scaled = Z;
```

La scala viene applicata solo alle coordinate planimetriche X e Y. Le quote Z non vengono modificate, rispettando l'affidabilità della componente verticale.

5) Creazione del nuovo nome file

```
[~, nameNoExt] = fileparts(filename);
suffix = ['_' num2str(round(scaleFactor*100)) '.txt'];
newFilename = fullfile(folder, [nameNoExt suffix]);
```

Il nuovo file mantiene il nome originale con l'aggiunta del suffisso indicante la scala applicata (es. _150.txt).

6) Salvataggio della nuvola scalata

```
fid = fopen(newFilename, 'w');
fprintf(fid, 'X(m) Y(m) Z(m)\n');
fprintf(fid, '%.3f %.3f %.3f\n', [X_scaled'; Y_scaled'; Z_scaled']);
fclose(fid);
disp(['Nuvola scalata salvata in: ' newFilename]);
```

Il file viene creato e popolato con le coordinate scalate. La stampa conferma il salvataggio all'utente.

Questo passaggio è fondamentale per integrare mappe storiche digitalizzate con rilievi moderni, perché le immagini possono avere scale non perfette a causa di fotocopie, scansioni o riduzioni del foglio originale. Con questa funzione, la nuvola viene corretta senza alterare la componente verticale, preservando le informazioni più affidabili delle mappe storiche.

Trasformazione (opzionale)

È possibile applicare una trasformazione rigida per allineare le coordinate digitalizzate a un sistema di riferimento esterno. Sono disponibili due modalità: traslazione e rototraslazione.

Traslazione: si selezionano due punti e se ne inseriscono le coordinate reali; viene applicato uno spostamento costante a tutta la nuvola di punti.

Rototraslazione: si selezionano tre punti con coordinate reali note; il software applica una rotazione e poi una traslazione per posizionare correttamente i dati in un sistema metrico esterno. Questa operazione fornisce un allineamento preliminare, utile per sovrapporre rapidamente i dati in ambiente GIS; un perfezionamento successivo rimane possibile.

```
% Selezione operazione
op = menu('Seleziona operazione', ...
    'Solo Traslazione', ...
    'Rototraslazione', ...
    'Annulla');

switch op
    case 1
        % Traslazione: inserimento coordinate note di due punti
        DX = x_real(1) - x_digi(1);
        DY = y_real(1) - y_digi(1);
        Xtrasl = X + DX;
        Ytrasl = Y + DY;
        Ztrasl = Z;
        save('nuvola_trasl.txt', 'Xtrasl', 'Ytrasl', 'Ztrasl', '-ascii');

    case 2
        % Rototraslazione con 3 punti noti
        % Calcolo matrice di trasformazione rigida
        [R, t] = rigid_transform_2D([x_digi', y_digi'], [x_real', y_real']);
```

```
punti_trasf = R * [X'; Y'] + t;  
Xtrasf = punti_trasf(1,:);  
Ytrasf = punti_trasf(2,:);  
Ztrasf = Z;  
save('nuvola_rtr.txt','Xtrasf','Ytrasf','Ztrasf','-ascii');  
end
```

APPENDICE 2

Script DigMap

In questo allegato è riportato l'intero codice modulare del software DigMap, nel formato degli script MATLAB utilizzati nel progetto. Il materiale è fornito esclusivamente ai fini di verifica e riproducibilità della metodologia descritta nel Rapporto Tecnico principale.

```
% Questo script Matlab si chiama DigMap ed è pensato per la digitalizzazione
di mappe storiche.
% Permette di caricare un'immagine raster e lavorare in modo interattivo con
il mouse per estrarre dati metrici e altimetrici.
% Il flusso operativo tipico è il seguente.
% Si carica la mappa e si leggono le coordinate X,Y cliccando con il mouse.
% Successivamente si definisce una scala metrica selezionando due punti noti
e indicando la distanza
% reale in metri.
% Da quel momento le linee tracciate sulla mappa possono essere digitalizzate
e salvate
% in file di testo, ognuna associata a una quota specifica.
% Lo script consente poi di importare i file di linee già salvati e
visualizzarli insieme con le quote.
% I dati possono essere interpolati su una griglia regolare per ottenere un
modello digitale del terreno
% (DEM), che può essere esportato come nuvola di punti tridimensionale.
% Sono previste anche funzioni opzionali. Una permette di applicare un
fattore di scala orizzontale a
% una nuvola di punti già esistente, utile per correzioni successive.
Un'altra consente di confrontare
% la nuvola originale e quella riscalata sia in due dimensioni sia come
superficie interpolata in forma di DEM.
% Il percorso di cartella è predefinito ma può essere modificato all'inizio
del codice in base alla
% propria configurazione. L'uso tipico richiede quindi di aprire un'immagine,
impostare la scala,
% digitalizzare le linee con quota, salvare i file, interpolare su griglia
e infine esportare la nuvola di
% punti o applicare le correzioni opzionali.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% DigMap - Matlab tool per digitalizzare le mappe storiche
% di Pesci e Randazzo INGVBO
% -----
% Questo script consente di:
% 1. Caricare immagini di mappe storiche.
% 2. Digitare linee e punti di quota con il mouse.
% 3. Definire una scala reale tra due punti.
% 4. Salvare i dati delle linee in file TXT strutturati.
% 5. Importare più linee e visualizzarle con le quote.
% 6. Interpolare i dati in una griglia regolare (DEM).
% 7. Esportare la nuvola di punti finale.
```

```

% 8. Opzionale - Applicare correzione di scala orizzontale alla nuvola.
% 9. Opzionale - Importare la nuvola originale e quella riscalata
% 10. Opzionale - Esportare la nuvola rototraslata con 3 punti noti (se
presenti)
% (bis) - Esportare la nuvola traslata (offset metrico)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% NB: in questo script ci sono folder e path iniziali predefiniti ('E:\mappe
storiche digitalizzate')
%% per adattare in toto il codice sostituire in base alla propria configurazione.

%% =====
%% 1. CARICAMENTO IMMAGINE E LETTURA COORDINATE CON MOUSE
%% =====

% --- Cartella iniziale di default ---
defaultFolder = 'E:\mappe storiche digitalizzate';

% --- Caricamento immagine ---
[fileName, folder] = uigetfile({'*.jpg;*.png;*.tif;*.bmp','Immagini (*.jpg,
*.png, *.tif, *.bmp)'}), ...
                                'Seleziona una immagine da visualizzare',
defaultFolder);
if isequal(fileName, 0)
    disp('Nessun file selezionato.');
```

return;

```

end

filePath = fullfile(folder, fileName);
img = imread(filePath);

% --- Ribaltamento verticale per origine in basso a sinistra ---
img = flipud(img);

% --- Visualizzazione ---
figure;
imshow(img);
set(gca, 'YDir', 'normal'); % Y cresce verso l'alto (coordinata cartesiana)
title('Posiziona il mouse sull''immagine per leggere le coordinate [X, Y]');
xlabel('X (pixel)');
ylabel('Y (pixel)');

% --- Attiva datacursormode con callback per mostrare coordinate ---
dcm_obj = datacursormode(gcf);
set(dcm_obj, 'Enable', 'on');
set(dcm_obj, 'DisplayStyle', 'window');

% Funzione anonima per mostrare coordinate X e Y al click col mouse
set(dcm_obj, 'UpdateFcn', @(src, event_obj) ...

```

```

        {'X: ', num2str(event_obj.Position(1))}, {'Y: ', num2str(event_obj.
Position(2))}]);

disp('Passa il mouse sull''immagine e clicca per leggere le coordinate (X,Y).
Premi ESC per uscire dalla modalità.');
```

```

%% =====
%% 2. DEFINIZIONE DELLA SCALA METRICA
%% =====

% --- Cartella iniziale di default ---
defaultFolder = 'E:\mappe storiche digitalizzate';

% --- Caricamento immagine ---
[fileName, folder] = uigetfile({'*.jpg;*.png;*.tif;*.bmp', 'Immagini (*.jpg,
*.png, *.tif, *.bmp)'}, ...
                                'Seleziona una immagine da visualizzare',
defaultFolder);
if isequal(fileName, 0)
    disp('Nessun file selezionato.');
```

```

    return;
end

filePath = fullfile(folder, fileName);
img = imread(filePath);

% --- Ribaltamento verticale per origine in basso a sinistra ---
img = flipud(img);

% --- Visualizzazione immagine ---
figure;
imshow(img);
set(gca, 'YDir', 'normal'); % Y cresce verso l'alto (cartesiano)
title('Clicca 2 punti sulla mappa per definire la scala spaziale');
xlabel('X (pixel)');
ylabel('Y (pixel)');
```

```

% --- Definizione scala ---
disp('Clicca 2 punti sulla mappa per definire la scala orizzontale (distanza
reale).');
```

```

[xScale, yScale] = ginput(2);

% Disegno linea fra i due punti scelti per riferimento visivo
hold on;
plot(xScale, yScale, 'ro-', 'LineWidth', 2);
hold off;

dPixel = sqrt( (xScale(2) - xScale(1))^2 + (yScale(2) - yScale(1))^2 );
dReal = input('Inserisci la distanza reale in metri tra i due punti selezionati:

```

```

');

scaleFactor = dReal / dPixel;
fprintf('Fattore di scala (metri/pixel): %.6f\n', scaleFactor);

% --- Inizio ciclo disegno linee ---
lineCount = 1;
filepath = folder; % directory per salvare i file

% --- Contenitori per salvataggio cumulativo ---
Xtot = [];
Ytot = [];
Ztot = [];

%%

while true
    disp(['Disegna la linea numero ' num2str(lineCount) '. Clicca punti, doppio click o invio per finire.']);
    figure(gcf); % porta la finestra corrente in primo piano

    try
        [x, y] = getline(gca, 'closed', false);
    catch ME
        fprintf('Errore durante getline: %s\n', ME.message);
        break;
    end

    if isempty(x)
        disp('Nessuna linea disegnata, esco. ');
        break;
    end

    quota = input('Inserisci quota per questa linea (default 0): ');
    if isempty(quota)
        quota = 0;
    end

    x_scaled = x * scaleFactor;
    y_scaled = y * scaleFactor;

    %rendiamo visibili I punti selezionati dopo avere finite la linea.
    figure(gcf); % assicurati che la figura sia attiva
    hold on;
    plot(x, y, '-o', 'Color', [0 0 1], 'LineWidth', 2, 'MarkerSize', 4);

    nPunti = length(x);
    quotaVec = quota * ones(nPunti, 1);
    dati = [x_scaled, y_scaled, quotaVec];

```

```

    % Nome file che include il numero linea e quota
    nomeFile = fullfile(filepath, sprintf('linea_%d_quota_%g.txt', lineCount,
    quota));

    fid = fopen(nomeFile, 'w');
    if fid == -1
        error(['Impossibile aprire il file: ' nomeFile]);
    end
    fprintf(fid, 'X(m) Y(m) Quota\n');
    fprintf(fid, '%f %f %f\n', dati');
    fclose(fid);

    disp(['Linea salvata in ' nomeFile]);

    %%%
    % --- Aggiungi i punti al dataset cumulativo ---
    Xtot = [Xtot; x_scaled];
    Ytot = [Ytot; y_scaled];
    Ztot = [Ztot; quotaVec];
    %%%

    lineCount = lineCount + 1;

    rispostaNum = input('Vuoi tracciare ancora una linea? (1 = sì, 0 = no):
    ');
    if isempty(rispostaNum) || rispostaNum ~= 1
        disp('Termino il processo. ');
        break;
    end
end

    %%%
    %% -----
    % Salvataggio opzionale file totale linee digitalizzate
    %% -----
    if ~isempty(Xtot)
        rispostaTot = input('Vuoi salvare un unico file con tutte le linee? (1
        = sì, 0 = no): ');
        if ~isempty(rispostaTot) && rispostaTot == 1
            nomeTotale = fullfile(filepath, 'tutte_le_linee.txt');
            fid = fopen(nomeTotale, 'w');
            fprintf(fid, 'X(m) Y(m) Z(m)\n');
            fprintf(fid, '%.3f %.3f %.3f\n', [Xtot'; Ytot'; Ztot']);
            fclose(fid);
            disp(['? File cumulativo salvato in: ' nomeTotale]);
        else
            disp('File cumulativo NON salvato. ');
        end
    end
end

```

```

%% =====
%% 3. DIGITAZIONE LINEE CON QUOTE - IMPORT
%% =====

defaultPath = 'E:\mappe storiche digitalizzate';

[files, path] = uigetfile(fullfile(defaultPath, '*.txt'), ...
    'Seleziona uno o più file TXT con linee', 'MultiSelect', 'on');

if isequal(files, 0)
    disp('Nessun file selezionato. ');
    return;
end

if ischar(files)
    files = {files};
end

figure;
hold on;
title('Linee importate con quote');
xlabel('X (m)');
ylabel('Y (m)');
axis equal;
grid on;

colormapLines = lines(length(files));

for i = 1:length(files)

    file = fullfile(path, files{i});
    data = importdata(file, ' ', 1); % salta la prima riga di header

    X = data.data(:,1);
    Y = data.data(:,2);
    Q = data.data(:,3);

    uniqueQ = unique(Q);

    for k = 1:length(uniqueQ)
        qVal = uniqueQ(k);

        idxQ = find(Q == qVal); % punti con la stessa quota

        % Identifica eventuali segmenti separati nella stessa quota
        gaps = find(diff(idxQ) > 1);
        segmentStarts = [1; gaps+1];
        segmentEnds    = [gaps; length(idxQ)];

        for s = 1:length(segmentStarts)

```

```

        seg = idxQ(segmentStarts(s):segmentEnds(s));

        % Plotta il segmento
        plot(X(seg), Y(seg), '-', 'Color', colormapLines(i,:), 'LineWidth',
1.5);

        % Posizione etichetta (centro del segmento)
        xLab = mean(X(seg));
        yLab = mean(Y(seg));

        text(xLab, yLab, num2str(qVal), ...
            'FontSize', 9, 'FontWeight', 'bold', ...
            'Color', colormapLines(i,:), ...
            'BackgroundColor', 'w', 'Margin', 1, ...
            'Clipping', 'on');
    end
end
end

hold off;

%% =====
%% 4. IMPORT E VISUALIZZAZIONE LINEE CON QUOTE
%% =====

% --- Selezione multipla file .txt ---
[files, folder] = uigetfile('E:\mappe storiche digitalizzate\*.txt', ...
    'Seleziona i file di quota (x,y,z)', 'MultiSelect', 'on');

if isequal(files, 0)
    disp('Nessun file selezionato.');
```

```

    return;
end

if ischar(files) % Selezionato un solo file (char), lo metto in cella
    files = {files};
end

% --- Caricamento dati ---
Xall = [];
Yall = [];
Zall = [];

for i = 1:length(files)
    filePath = fullfile(folder, files{i});
    data = dlmread(filePath, '\', 1, 0); % skip header 1 riga
    Xall = [Xall; data(:,1)];
    Yall = [Yall; data(:,2)];
    Zall = [Zall; data(:,3)];
end

```

```

end

% --- Chiedi estensione della griglia e risoluzione ---
prompt = {'xmin:', 'xmax:', 'ymin:', 'ymax:', 'passo griglia (m):'};
dlgtitle = 'Definisci limiti e risoluzione griglia';
dims = [1 35];
definput = {num2str(min(Xall)), num2str(max(Xall)), num2str(min(Yall)),
num2str(max(Yall)), '1'};
answer = inputdlg(prompt, dlgtitle, dims, definput);

if isempty(answer)
    disp('Input annullato. ');
    return;
end

xmin = str2double(answer{1});
xmax = str2double(answer{2});
ymin = str2double(answer{3});
ymax = str2double(answer{4});
step = str2double(answer{5});

%% =====
%% 5. INTERPOLAZIONE SU GRIGLIA (DEM)
%% =====

xq = xmin:step:xmax;
yq = ymin:step:ymax;
[Xq, Yq] = meshgrid(xq, yq);

% --- Interpolazione con metodo 'natural' ---
Zq = griddata(Xall, Yall, Zall, Xq, Yq, 'natural');

% --- Chiedi valori min/max colorbar ---
promptC = {'Valore minimo scala colori:', 'Valore massimo scala colori:'};
dlgtitleC = 'Imposta scala colori';
definputC = {num2str(nanmin(Zq(:))), num2str(nanmax(Zq(:)))};
answerC = inputdlg(promptC, dlgtitleC, dims, definputC);

if isempty(answerC)
    disp('Input scala colori annullato. ');
    return;
end

cmin = str2double(answerC{1});
cmax = str2double(answerC{2});

% --- Plot 2D del DEM interpolato ---
figure;
imagesc(xq, yq, Zq);

```

```

set(gca, 'YDir', 'normal');
colormap(jet); % Puoi anche provare 'parula', 'turbo', o 'coolwarm' se hai
pacchetti extra
caxis([cmin cmax]);
colorbar;
title('DEM interpolato con punti originali');
xlabel('X (m)');
ylabel('Y (m)');
axis equal tight;
hold on;

% --- Plotta i punti originali ---
scatter(Xall, Yall, 15, Zall, 'filled', 'MarkerEdgeColor', 'k');
% scatter usa la stessa scala colori per Zall automaticamente
% MarkerEdgeColor nero per evidenziare i punti

% --- Opzioni estetiche ---
set(gca, 'FontSize', 12);

hold off;

%% =====
%% 6. SALVATAGGIO NUVOLA DI PUNTI
%% =====

% Trasformiamo le matrici in vettori colonna
Xv = Xq(:);
Yv = Yq(:);
Zv = Zq(:);

% Troviamo gli indici dove Z non è NaN
validIdx = ~isnan(Zv);

% Filtriamo i vettori per tenere solo i valori validi
Xv_valid = Xv(validIdx);
Yv_valid = Yv(validIdx);
Zv_valid = Zv(validIdx);

% Nome file output completo con percorso
outputFile = fullfile(folder, 'nuvola.txt');

% Apri file per scrittura
fid = fopen(outputFile, 'w');
if fid == -1
    error('Impossibile aprire il file per la scrittura.');
```

```

end

% Scrivi intestazione
fprintf(fid, 'X(m) Y(m) Z(m)\n');
```

```

% Scrivi dati validi con 3 cifre decimali
fprintf(fid, '%.3f %.3f %.3f\n', [Xv_valid'; Yv_valid'; Zv_valid']);

% Chiudi file
fclose(fid);

disp(['Nuvola di punti salvata in: ' outputFile]);

%% =====
%% 7. SCALA ORIZZONTALE SU NUVOLE ESISTENTI
%% =====

% --- Carica file nuvola originale ---
[filename, folder] = uigetfile('E:\mappe storiche digitalizzate\*.txt',
'Seleziona il file nuvola originale');
if isequal(filename,0)
    disp('Nessun file selezionato. ');
    return;
end

filepath = fullfile(folder, filename);

% Salta intestazione, legge dati
data = dlmread(filepath, '', 1, 0); % Assume 1 riga header
X = data(:,1);
Y = data(:,2);
Z = data(:,3);

% Chiedi fattore di scala (percentuale assoluta)
prompt = {'Inserisci la scala orizzontale in percentuale (es. 200 per doppio,
50 per metà)'};
dlgtitle = 'Scala orizzontale';
dims = [1 50];
definput = {'100'};
answer = inputdlg(prompt, dlgtitle, dims, definput);

if isempty(answer)
    disp('Operazione annullata. ');
    return;
end

scaleFactor = str2double(answer{1}) / 100;
% Applica la scala solo su X e Y
X_scaled = X * scaleFactor;
Y_scaled = Y * scaleFactor;
Z_scaled = Z; % invariato

% Costruisci nome file output corretto

```

```

[~, nameNoExt] = fileparts(filename); % prendi solo nome senza estensione e
path
suffix = ['_' num2str(round(scaleFactor*100)) '.txt'];
newFilename = fullfile(folder, [nameNoExt suffix]);

disp(['Provo a salvare in: ' newFilename]);

% Prova ad aprire il file in scrittura
fid = fopen(newFilename, 'w');
if fid == -1
    error(['Impossibile aprire il file per scrittura: ' newFilename]);
end

fprintf(fid, 'X(m) Y(m) Z(m)\n');
fprintf(fid, '%.3f %.3f %.3f\n', [X_scaled'; Y_scaled'; Z_scaled']);
fclose(fid);
disp(['Nuvola scalata salvata in: ' newFilename]);

%% =====
% 8. Opzionale - Applicare correzione di scala orizzontale alla nuvola.
%% =====

% --- Visualizza nuvola originale e scalata in 2D (X,Y) ---
% Chiedi sottocampionamento
prompt = {'Inserisci passo di sottocampionamento per la visualizzazione (es.
1 = tutti, 10 = un punto ogni 10)'};
dlgtitle = 'Sottocampionamento visualizzazione';
dims = [1 50];
definput = {'1'};
answer = inputdlg(prompt, dlgtitle, dims, definput);

if isempty(answer)
    disp('Visualizzazione annullata. ');
    return;
end

step = str2double(answer{1});
if isnan(step) || step < 1
    step = 1; % default
end

% Applica sottocampionamento
X_sub = X(1:step:end);
Y_sub = Y(1:step:end);
X_scaled_sub = X_scaled(1:step:end);
Y_scaled_sub = Y_scaled(1:step:end);

% Plot 2D
figure;

```

```

plot(X_sub, Y_sub, '.r'); hold on;
plot(X_scaled_sub, Y_scaled_sub, '.b');
hold off;
xlabel('X (m)');
ylabel('Y (m)');
legend('Originale','Scalata');
title(['Confronto 2D (sottocampionamento 1 ogni ' num2str(step) ')']);
axis equal;
grid on;

%% =====
% 9. Opzionale - Importare la nuvola originale e quella riscalata
%% =====

% --- Costruzione DEM e visualizzazione ---
% Definisci griglia regolare basata sui limiti della nuvola
gridRes = 1; % passo griglia in metri (modifica se vuoi più dettaglio)
% Limiti originali
xlim = [min(X) max(X)];
ylim = [min(Y) max(Y)];
[xq,yq] = meshgrid(xlim(1):gridRes:xlim(2), ylim(1):gridRes:ylim(2));

% Interpolazione DEM originale
Zq_orig = griddata(X, Y, Z, xq, yq, 'natural');

% Interpolazione DEM scalata
Zq_scaled = griddata(X_scaled, Y_scaled, Z_scaled, xq, yq, 'natural');

% Figura con 1 riga, 2 colonne
figure;
subplot(1,2,1);
imagesc(xlim, ylim, Zq_orig);
axis equal tight;
set(gca,'YDir','normal'); % mantiene asse Y normale (non invertito)
colormap jet; colorbar;
title('DEM Originale');
subplot(1,2,2);
imagesc(xlim, ylim, Zq_scaled);
axis equal tight;
set(gca,'YDir','normal');
colormap jet; colorbar;
title('DEM Scalato');

%% =====
%% 10. Opzionale - Allineamento metrico tramite punti di controllo
%% =====

choice = questdlg('Vuoi applicare un allineamento metrico tramite GCP
esterni?', ...

```

```

        'Allineamento opzionale', 'Si', 'No', 'No');

if strcmp(choice, 'Si')

    usaMemoria = false;

    % Verifica se la nuvola è già in memoria
    if exist('Xv_valid', 'var') && exist('Yv_valid', 'var') && exist('Zv_
valid', 'var') ...
        && ~isempty(Xv_valid)

        sceltaNuvola = questdlg( ...
            'Vuoi usare la nuvola già generata oppure caricarne una nuova?',
            ...
            'Scelta nuvola da allineare', ...
            'Usa memoria', 'Carica file', 'Carica file');

        if strcmp(sceltaNuvola, 'Usa memoria')
            usaMemoria = true;
        end
    end

    if ~usaMemoria
        [fileNuvola, pathNuvola] = uigetfile(fullfile(folder, '*.txt'), ...
            'Seleziona il file della nuvola da allineare');
        if isequal(fileNuvola, 0)
            disp('Nessun file selezionato ? salto modulo.');
```

return;

```

        end
        nuvolaFile = fullfile(pathNuvola, fileNuvola);
        data = importdata(nuvolaFile, '\ ', 1);
        Xv_valid = data.data(:,1);
        Yv_valid = data.data(:,2);
        Zv_valid = data.data(:,3);
    end

    %% Selezione GCP con feedback
    disp('Seleziona i punti GCP sulla nuvola ? poi premi INVIO per terminare');
    figGCP = figure('Name', 'Seleziona punti GCP');
    scatter(Xv_valid, Yv_valid, 10, 'filled');
    title('Seleziona GCP con click ? premi INVIO per terminare');
    xlabel('X (m)'); ylabel('Y (m)');
    axis equal; grid on;
    zoom off; pan off; rotate3d off;
    figure(gcf);

    [x_i, y_i] = ginput; % termina con INVIO

    % Filtra eventuali click nulli
    x_i = x_i(~isnan(x_i));

```

```

y_i = y_i(~isnan(y_i));

nGCP = length(x_i);

if nGCP == 0
    disp('Nessun GCP selezionato ? modulo annullato.');
```

close(figGCP);

```

    return;
end

if nGCP < 2
    error('Servono almeno 2 GCP per definire una trasformazione.');
```

close(figGCP);

```

    return;
end

hold on;
plot(x_i, y_i, 'ro', 'MarkerSize', 10, 'LineWidth', 2);
hold off;
drawnow;

%% Inserimento coordinate target
disp('Inserisci le coordinate target dei GCP:');
X_target = zeros(nGCP,1);
Y_target = zeros(nGCP,1);

for k = 1:nGCP
    X_target(k) = str2double(inputdlg(['X target punto ',num2str(k)],
    'Input GCP', 1));
    Y_target(k) = str2double(inputdlg(['Y target punto ',num2str(k)],
    'Input GCP', 1));
end

%% Trasformazione: rot + scala + trasl
inputPoints = [x_i y_i];
targetPoints = [X_target Y_target];
tform = fitgeotrans(inputPoints, targetPoints, 'nonreflectivesimilarity');

[X_al, Y_al] = transformPointsForward(tform, Xv_valid, Yv_valid);
Z_al = Zv_valid; % Quota invariata

%% Salvataggio risultato
outFile = fullfile(folder, 'nuvola_allineata.txt');
fid = fopen(outFile,'w');
fprintf(fid,'X(m) Y(m) Z(m)\n');
fprintf(fid,'%0.3f %0.3f %0.3f\n',[X_al'; Y_al'; Z_al']);
fclose(fid);

figure; scatter(X_al, Y_al, 10, Z_al, 'filled');
axis equal; grid on; colorbar;

```

```

title('Nuvola allineata tramite GCP esterni');
xlabel('X (m)'); ylabel('Y (m)');

msgbox(['Allineamento completato ? File salvato: ', outFile]);
end

%% =====
%% 10bis. Opzionale - Offset metrico noto (solo traslazione)
%% =====
choice2 = questdlg('Vuoi applicare un offset di traslazione noto?', ...
    'Offset metrico', 'Si', 'No', 'No');

if strcmp(choice2, 'Si')

    usaMemoria2 = false;

    if exist('Xv_valid', 'var') && exist('Yv_valid', 'var') && exist('Zv_
valid', 'var') ...
        && ~isempty(Xv_valid)

        sceltaNuvola = questdlg( ...
            'Vuoi usare la nuvola già generata oppure caricarne una nuova?',
...
            'Scelta nuvola per offset', ...
            'Usa memoria', 'Carica file', 'Carica file');

        if strcmp(sceltaNuvola, 'Usa memoria')
            usaMemoria2 = true;
        end
    end

    if ~usaMemoria2
        [fileNuvola, pathNuvola] = uigetfile(fullfile(folder, '*.txt'), ...
            'Seleziona il file della nuvola da traslare');
        if isequal(fileNuvola, 0)
            disp('Operazione annullata.');
```

return;

```

        end
        nuvolaFile = fullfile(pathNuvola, fileNuvola);
        data = importdata(nuvolaFile, '\n', 1);
        Xv_valid = data.data(:,1);
        Yv_valid = data.data(:,2);
        Zv_valid = data.data(:,3);
    end

    prompt = {'Offset X (m):', 'Offset Y (m):'};
    answer = inputdlg(prompt, 'Inserisci offset assoluto', [1 40], {'0', '0'});
    if isempty(answer), msgbox('Operazione annullata.');
```

return; end

```

offsetX = str2double(answer{1});
offsetY = str2double(answer{2});

X_off = Xv_valid + offsetX;
Y_off = Yv_valid + offsetY;
Z_off = Zv_valid;

% Salvataggio
outFile2 = fullfile(folder, 'nuvola_offset.txt');
fid2 = fopen(outFile2, 'w');
fprintf(fid2, 'X(m) Y(m) Z(m)\n');
fprintf(fid2, '%.3f %.3f %.3f\n', [X_off'; Y_off'; Z_off']);
fclose(fid2);

figure;
scatter(X_off, Y_off, 10, Z_off, 'filled');
axis equal; grid on; colorbar;
title('Nuvola con offset metrico applicato');
xlabel('X (m)'); ylabel('Y (m)');

msgbox(['Offset applicato. File salvato: ', outFile2]);
end

```




QUADERNI di GEOFISICA

ISSN 1590-2595

<https://collane-editoriali.ingv.it/index.php/qdg/it/index/>

QUADERNI di GEOFISICA è una rivista scientifica che pubblica dati, misurazioni e osservazioni geofisiche (anche preliminari) in italiano e inglese. L'obiettivo è una diffusione rapida e gratuita per la comunità scientifica, garantita da un processo di peer-review e da un Editorial Board multidisciplinare. La rivista è inclusa in "Emerging Sources Citation Index" di Clarivate Analytics e "Open Access Journals" di Scopus, e adotta il modello Diamond Open Access, che non prevede costi né per gli autori né per i lettori.

QUADERNI di GEOFISICA is a scientific journal that publishes geophysical data, measurements and observations, including preliminary results, in Italian and English. Its aim is to ensure rapid and free dissemination to the research community, supported by a peer-review process and by a multidisciplinary Editorial Board. The journal is included in the Emerging Sources Citation Index of Clarivate Analytics and it's included in the Open Access journals of Scopus. It follows the Diamond Open Access model, with no charges for either authors or readers.



RAPPORTI TECNICI INGV

ISSN 2039-7941

<https://collane-editoriali.ingv.it/index.php/rt/it/index>

RAPPORTI TECNICI INGV è una rivista tecnico-scientifica che mette in mostra le numerose attività tecnologiche e di ricerca condotte dai ricercatori e tecnici INGV, e non solo. Pubblicata in italiano e inglese, RAPPORTI TECNICI INGV è ideata per valorizzare anche quei lavori che offrono un contributo significativo dal punto di vista tecnico, operativo o documentale fondamentale per lo sviluppo dell'Ente. La rivista adotta il modello Diamond Open Access, che non prevede costi né per gli autori né per i lettori.

RAPPORTI TECNICI INGV is a technical-scientific journal that showcases the wide range of technological and research activities carried out by INGV researchers and technical staff, as well as by external contributors. Published in Italian and English, RAPPORTI TECNICI INGV is designed to highlight works that provide a significant contribution from a technical, operational, or documentary perspective, and that are essential to INGV development. The journal follows the Diamond Open Access model, with no charges for either authors or readers.



MISCELLANEA INGV

ISSN 2039-6651

<https://collane-editoriali.ingv.it/index.php/mi/it/index>

MISCELLANEA INGV rappresenta una tipologia di editoria non tradizionalmente catalogabile nelle ordinarie riviste scientifiche di settore. Pubblicata in lingua italiana e inglese, MISCELLANEA INGV offre uno spazio per la pubblicazione di materiali eterogenei che, pur non rientrando nei formati più strutturati delle altre Collane editoriali INGV, rivestono un interesse scientifico, tecnico o divulgativo. La rivista adotta il modello Diamond Open Access, che non prevede costi né per gli autori né per i lettori.

MISCELLANEA INGV represents a form of publishing that cannot be classified within the standard scientific journals. Published in Italian and English, MISCELLANEA INGV provides a venue for the publication of heterogeneous materials that are of scientific, technical or outreach interest, even though they do not fall within the more structured formats of the other INGV editorial series. The journal follows the Diamond Open Access model, with no charges for either authors or readers.

Progetto grafico
Barbara ANGIONI
Istituto Nazionale di Geofisica e Vulcanologia

Ufficio Editoriale
Francesca DI STEFANO | Coordinatore

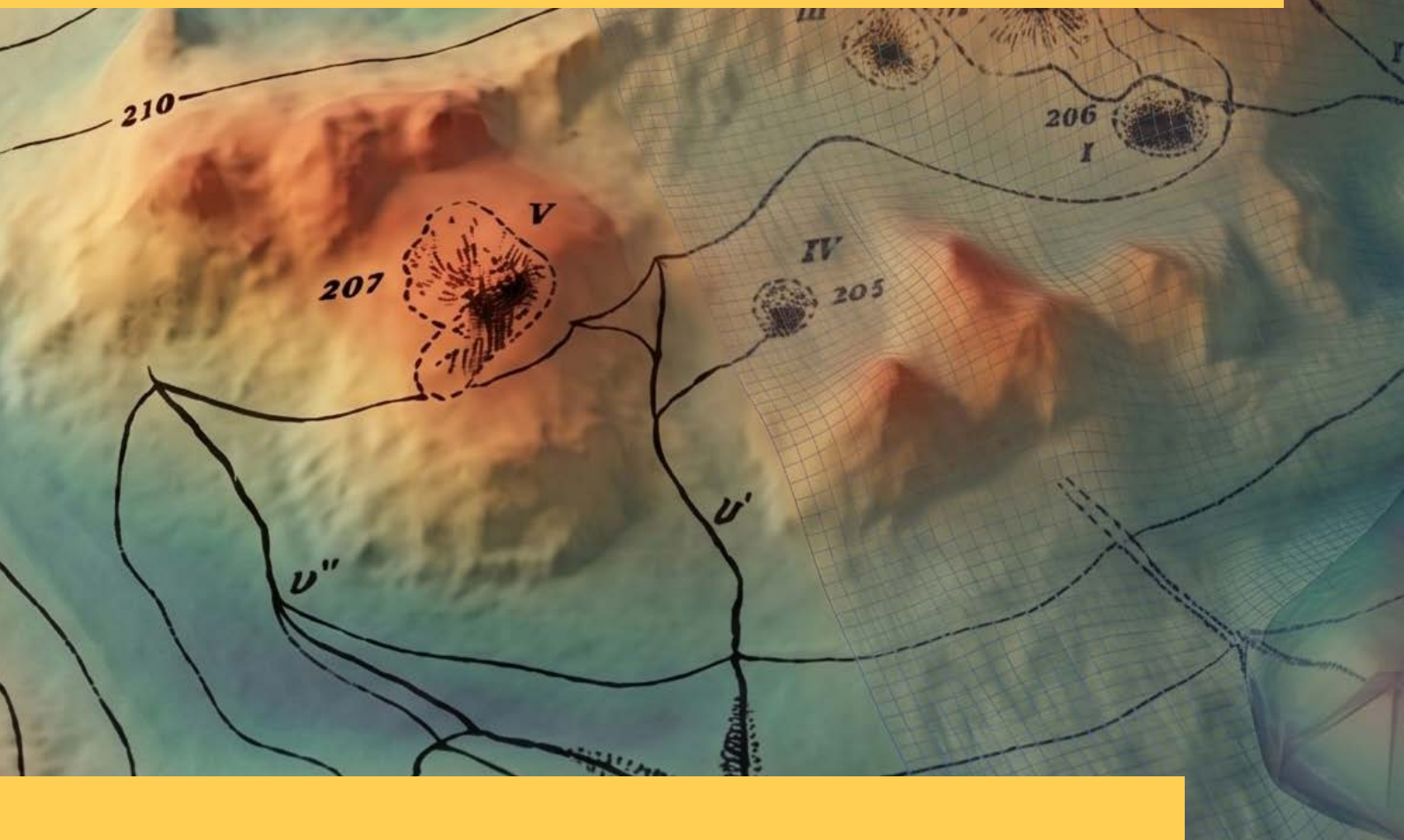
Segreteria di redazione
segreteria.collane-editoriali@ingv.it

Editing | Produzione | Grafica redazionale
Barbara ANGIONI
Massimiliano CASCONI
Francesca DI STEFANO
Patrizia PANTANI

© 2026 the Author(s). All rights reserved. Open Access



Creative Commons Attribution 4.0 International



ISTITUTO NAZIONALE DI GEOFISICA E VULCANOLOGIA