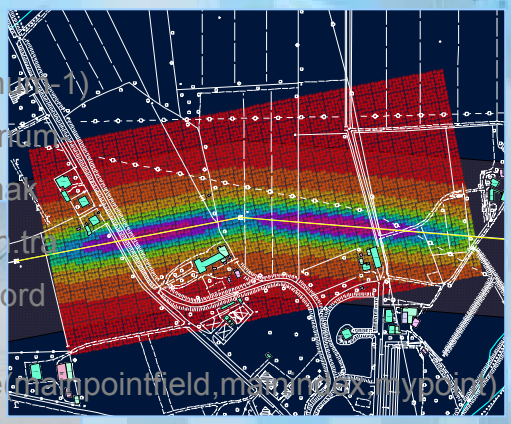
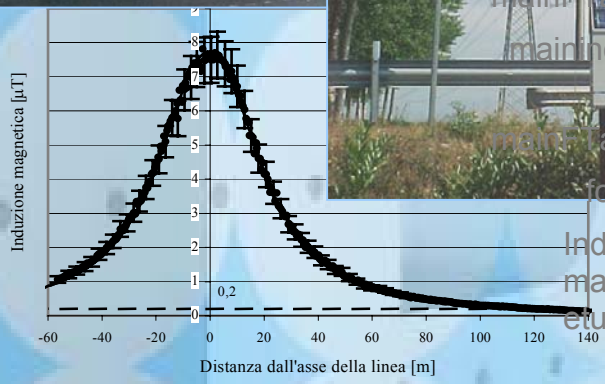
Agenzia Regionale per la Protezione dell'Ambiente del Friuli Venezia Giulia

ELETTRODOTTI SUL TERRITORIO DELLA PROVINCIA DI UDINE
VALUTAZIONE D'IMPATTO SULLA POPOLAZIONE



```
mainindex=0
for each m in 0..(xnum-1)
 for each n in 0..(ynum
  point_mn=point.mak
  mypoint=point_mn.tra
  mainFTab.addrecord
  mainFTab.setvalue(mainpointfield,mainindex,x,y,pointm)
  mainindex=mainindex+1
  mainFTab.setvalue(mainb.get(nnb),nnindex,indmagn)
  for each nny in 0..(ynum-1)
   Indmagn=If.readelt.asnumber
   mainFTab.setvalue(mainfieldx,nnindex,mainftab.r
   eturnvalue(mainpointfield,nnindex).getx)
```



Allegato all'elaborato conclusivo della seconda fase del progetto:
Valutazione dei livelli di campo magnetico prodotti dagli elettrodotti e del numero di edifici presenti all'interno di determinate fasce di rispetto

RESPONSABILE DEL PROGETTO
Renato Villalta

REFERENTE
Massimo Telesca

ELABORAZIONE
Francesco Montanari
Massimiliano Benes
Moreno Comelli
Alessandra Petrini
Luca Piani
Francesca Tassan Mazzocco

"Elettrodotti sul territorio della Provincia di Udine. Valutazione dell'impatto sulla popolazione."

Allegato all'elaborato conclusivo della seconda fase del progetto:

Valutazione dei livelli di campo magnetico prodotti dagli elettrodotti e del numero di edifici presenti all'interno di prefissate zone di rispetto

Indice

1. Introduzione

L'utilizzo di software previsionali per la valutazione del campo di induzione magnetica prodotto dagli elettrodotti è previsto dalla normativa nazionale [1], della guida CEI di riferimento [2] e dalle Linee Guida ANPA [3].

ARPA FVG ha acquisito a tal fine presso l'APPA di Trento il programma CMagnetico [4,5], realizzato dall'Istituto Trentino di Cultura e presente nell'elenco dei software previsionali accettati da ANPA.

La consistenza fra CMagnetico e la strumentazione e procedure di misura adottate da ARPA FVG è stata verificata in una campagna di misure organizzata a tale scopo, come descritto al paragrafo 2.

CMagnetico è stato quindi integrato con il catasto degli elettrodotti, costituito nelle fasi precedenti di questo studio, realizzando un apposito pacchetto di routines che gestisce il trasferimento di dati fra l'ambiente GIS ARCVIEW 3.2 ed il software previsionale, come descritto nel paragrafo 3.

Dopo l'analisi della normativa regionale della Regione Veneto [6] e di altre fonti [7], si sono elaborati dei criteri per la definizione di fasce di rispetto tali da ritenere che il livello del campo di induzione magnetica all'esterno di tali fasce sia inferiore a 0.2μT, come descritto nel paragrafo 4.

Va osservato che, sebbene il D.P.C.M. 8-7-2003 fissi l'obiettivo di qualità per l'intensità del campo di induzione magnetica a 3μT, si è preferito in questa fase del presente lavoro mantenere la soglia pari al valore previsto dalla succitata normativa regionale ed assunto come *soglia epidemiologica* in diversi studi [8].

La definizione delle fasce di attenzione per le varie linee ha consentito di individuare gli edifici effettivamente interessati (paragrafo 5), dopo che nelle fasi precedenti del progetto si era provveduto a ricavare dalla Carta Tecnica Regionale Numerica (CTRN) tutte le costruzioni distanti meno di 200m dall'asse degli elettrodotti.

2. Validazione del software previsionale CMagnetico

Nel caso di elettrodotti costituiti da un'unica linea, in assenza di altre sorgenti ELF vicine, il valore dell'induzione magnetica in prossimità della linea risulta direttamente proporzionale all'intensità di corrente circolante.

In questo caso, ottenuti dall'ente gestore i dati di corrente istantanea, è possibile stimare la consistenza tra i valori misurati di induzione magnetica e le simulazioni realizzate utilizzando un software previsionale.

Con l'ausilio del database degli elettrodotti, sviluppato nell'ambito di questo stesso studio, è stato possibile individuare sulla CTRN due linee a 380 kV, idonee per questo tipo di test. In Figura 1 viene mostrata la linea 21347 Planais-Salgareda, presso il sostegno 142, mentre in Figura 2 è rappresentata la linea 21356 Planais-Redipuglia, in prossimità del sostegno 160.

Sono state eseguite due misure di profilo dell'induzione magnetica trasversalmente alle linee utilizzando il sistema di misura EMDEX II nella versione LINDA, in cui lo strumento palmare viene fissato su di una rotella munita di contagiri

[9]. Nelle Figure 1 e 2 viene mostrato sia il percorso sulla CTRN che il confronto, per il profilo dell'induzione magnetica, tra le misure ed il calcolo effettuato con CMagnetico.

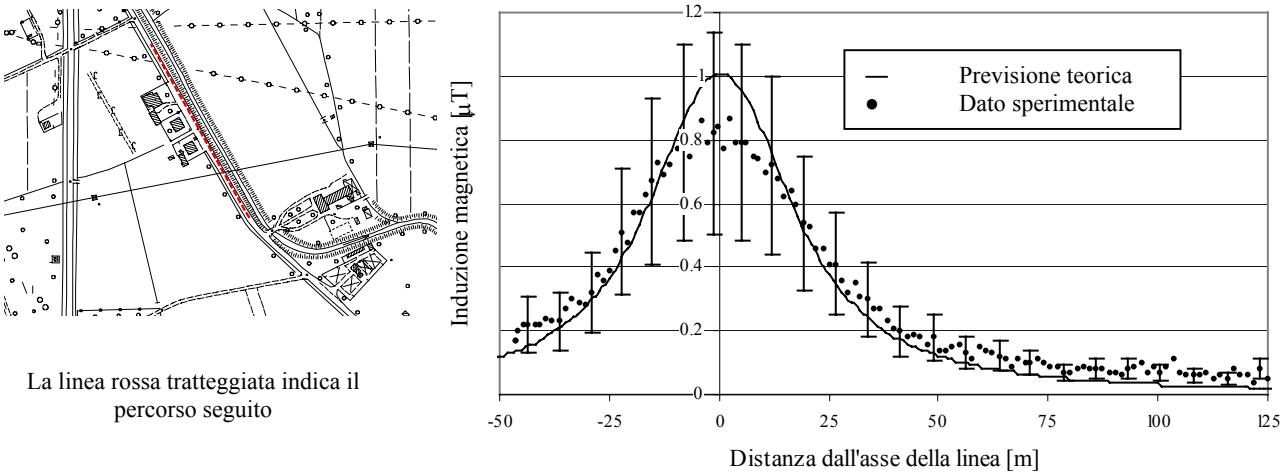


Figura 1: Confronto tra dati sperimentali e previsione teorica: Linea 21347 (380 kV, Planais –Salgareda) a San Giorgio di Nogaro (UD).

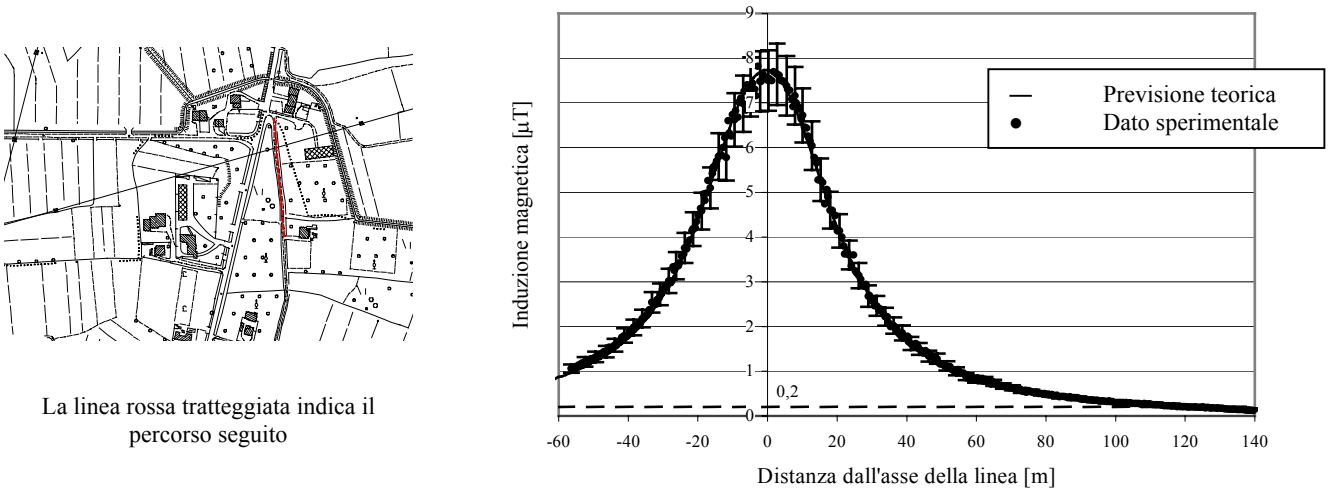


Figura 2: Confronto tra dati sperimentali e previsione teorica: Linea 21356 (380 kV, Planais–Redipuglia) a Cervignano del Friuli (UD)

È anche possibile desumere dai grafici l'estensione della fascia con valori superiori agli 0,2 µT. Su tali dati è stato fatto uno studio del chi quadrato ridotto $\tilde{\chi}^2$ [10]. È stato assunto un errore sperimentale del 3%, come riportato nel manuale d'uso dello strumento, al quale viene sommata in quadratura la fluttuazione percentuale di corrente, ricavabile dai dati forniti dall'ente gestore. Nel caso della linea 21347, per un valore della corrente di 130 ± 50 A (a fronte di una corrente media di 284 A e 95° percentile di 736 A nell'anno 2002), il valore del $\tilde{\chi}^2$ con 84 gradi di libertà risulta pari a 1,0. Nel caso della linea 21356, per un valore della corrente di 1110 ± 90 A (a fronte di una corrente media di 899 A e 95° percentile di 1811 A nell'anno 2002), è stato ottenuto un $\tilde{\chi}^2$ con 200 gradi di libertà pari a 0,6. In nessun caso il valore del $\tilde{\chi}^2$ ottenuto comporta il rigetto dell'ipotesi di consistenza fra le misure e la simulazione, fissata al 5% la soglia di rigetto.

3. Integrazione del software previsionale con il catasto degli elettrodotti in ambiente GIS

Al fine di agevolare e rendere più efficaci le campagne di misura previste nella terza fase di questo studio, si è ritenuto opportuno integrare CMagnetico con il catasto degli elettrodotti. Il catasto realizzato, sebbene presenti una struttura delle tabelle semplificata rispetto al modello proposto da ANPA [11], rispetto a questo mantiene intatto il contenuto di informazione, che è tale da consentire l'effettuazione delle simulazioni quando:

- non vi sia sovrapposizione significativa del campo prodotto da linee indipendenti: essendovi infatti uno sfasamento arbitrario fra le correnti circolanti, il campo magnetico risultante è imprevedibile;
- i proprietari ed i gestori degli elettrodotti abbiano risposto in modo esauriente alle richieste di questa Agenzia.

Il problema dell'integrazione desiderata si configura nei termini seguenti:

1) esportazione dall'ambiente GIS dei dati necessari all'effettuazione della simulazione con CMagnetico;
2) importazione nell'ambiente GIS dei risultati della simulazione effettuata con CMagnetico;

3.1 Esportazione dall'ambiente GIS dei dati necessari all'effettuazione della simulazione con CMagnetico

CMagnetico prevede normalmente l'inserimento manuale dei dati (valori di corrente, geometria della campate e dei tralicci) in un'apposita schermata; tali dati vengono quindi salvati in un file con estensione *.lcm, dopo una opportuna elaborazione, per poter essere caricati nuovamente in un altro modulo del programma e costitutire la base per una simulazione.

L'esportazione automatica dei dati avviene con la creazione del file *.lcm direttamente da ArcView.

Dopo che sul display sono state selezionate le campate per le quali si intende effettuare la simulazione, può essere lanciato lo script Avenue *Simul_launch.ave* di ArcView, che esegue le seguenti operazioni principali:

1.a) legge i dati delle campate selezionate nella tabella associata al file Campate.shp (corrente media, parametro della catenaria, numero di linea, codice dei sostegni precedenti e seguenti ciascuna campata) ed i dati dei sostegni corrispondenti nella tabella associata a Sostegni.shp (coordinate Gauss-Boaga, quote altimetriche d'attacco dei conduttori, sbracci delle mensole, lunghezza degli isolatori);

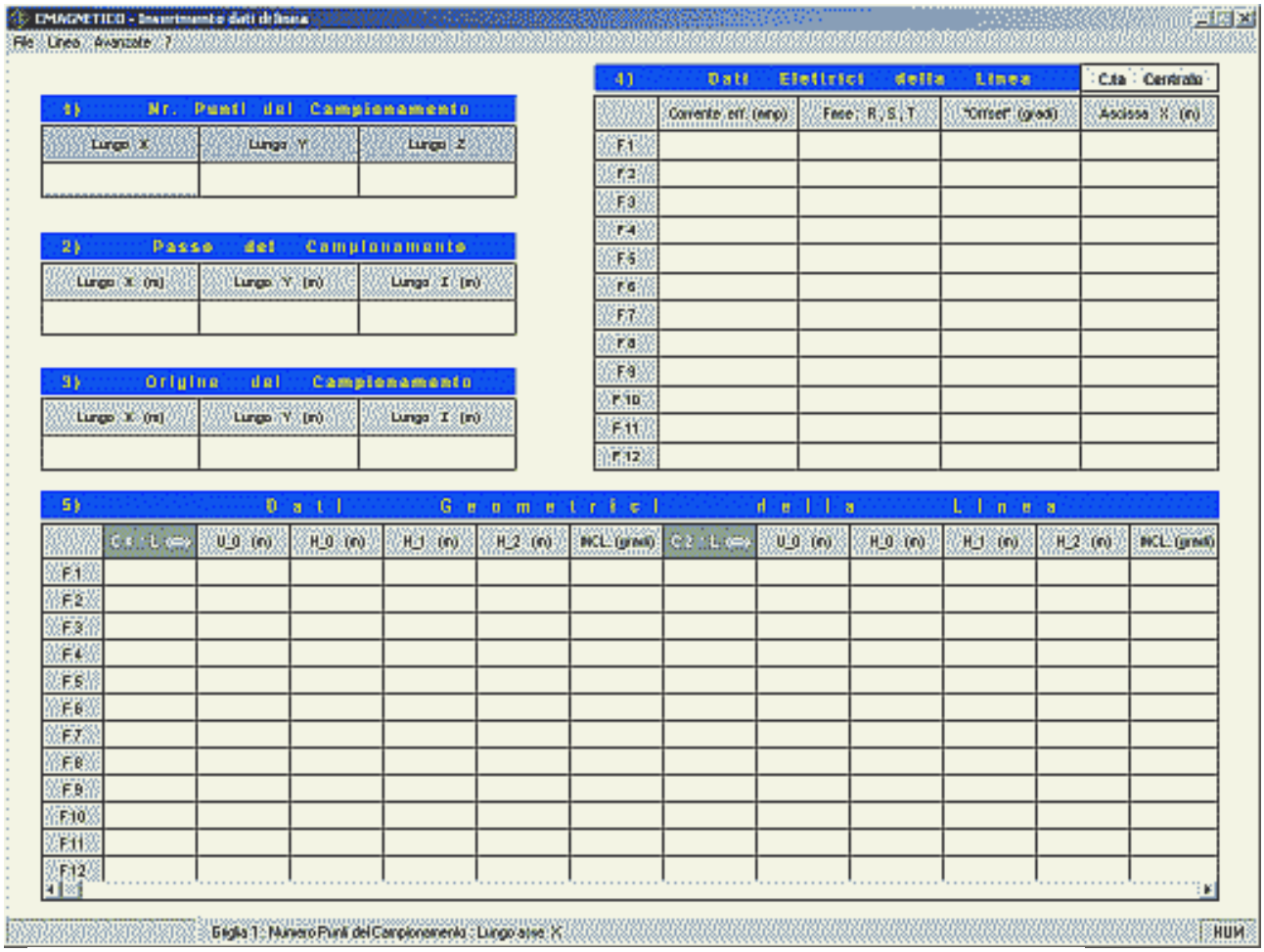


Figura 3: schermata di CMagnetico destinata all'inserimento dei dati delle campate.

1.b) richiede l'inserimento dei parametri di campionamento da utilizzare per la simulazione (numero di punti nelle tre direzioni; distanza fra i campioni nelle tre direzioni; origine del reticolo di simulazione);

1.c) elabora i dati reperiti nelle tabelle (ad esempio: calcola le coordinate e la quota del punto di minimo della catenaria per ciascuna campata; definisce il sistema di riferimento per la simulazione e ricalcola le coordinate di tutti gli oggetti interessati in funzione di esso);

1.d) genera un file di testo ASCII, richiedendone interattivamente il nome all'utente, contenente i dati elaborati;

1.e) lancia il codice Java che traduce il file ASCII in un file *.lcm in cui i dati sono scritti in forma binaria;

I dettagli di quanto realizzato a questo riguardo sono illustrati i

Figura 5: nel grafico, estensione delle fasce di rispetto in funzione della corrente circolante, per le tipologie di linea individuate. Nel caso del tipo *132kV;singola terna*, viene evidenziata la sovrapposizione fra l'andamento ricavato dal calcolo con CMagnetico e la funzione assunta a modello. Nel disegno, le considerazioni che portano alla scelta del modello.

3.2 Importazione nell'ambiente GIS dei risultati della sim

Dopo l'esecuzione della simulazione, CMagnetico scrive un fi ne. Simul_launch.ave esegue quindi le operazioni seguenti:

2.a) genera uno ShapeFile di Arcview di tipo puntua ne, con dimensioni, numero di punti e georeferenziazione opp

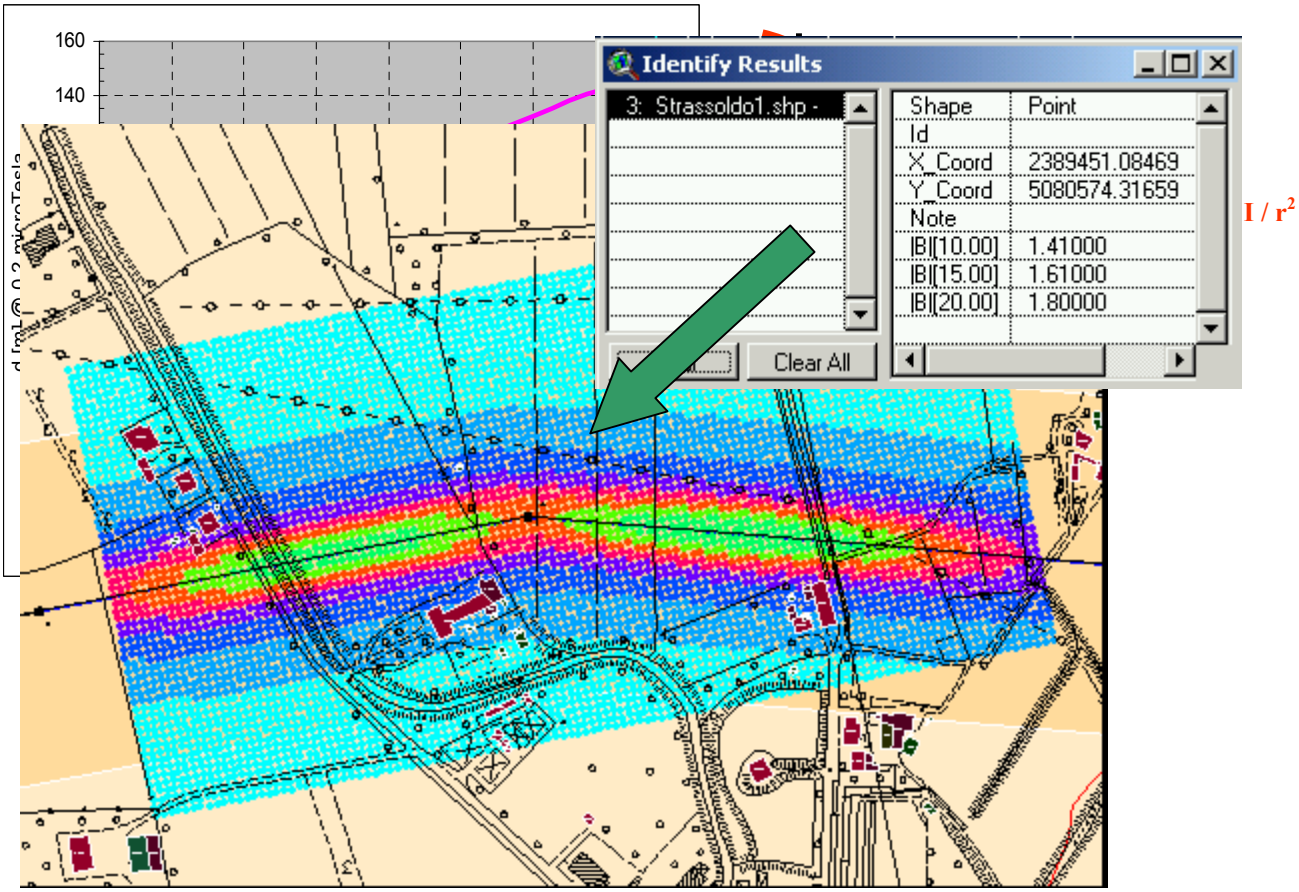
2.b) attribuisce a ciascun punto dello ShapeFile precedente i valori di campo simulati alle diverse quote;

I dettagli di quanto realizzato a questo riguardo sono illustrati in Appendice A.

4. Studio dei criteri per la determinazione delle fasce di attenzione

Si intende determinare delle fasce di attenzione tali che i valori di campo esternamente a tali fasce siano inferiori di una soglia prefissata.

Per ottenere un modello applicabile a tutte le linee, si ricorre ad alcune ipotesi semplificative e cautelative:



- i conduttori sono considerati essere perfettamente tesi;

Figura 4: simulazione del campo di induzione magnetica, georeferenziata e visualizzata in ambiente ArcViewGIS per mezzo dell'integrazione fra catasto degli elettrodotti e CMagnetico. La tabella evidenzia come, ad ogni punto della simulazione, siano associati i valori del campo alle diverse quote richieste.

- le campate infinitamente lunghe e rettilinee;
- si individuano 6 tipologie di linea – intesa come disposizione geometrica delle mensole e, dunque, dei punti di sospensione dei conduttori.

La procedura adottata si basa sull'osservazione che il modulo del campo di induzione magnetica generato da una singola terna equilibrata ha un andamento approssimabile ad $1/r^2$, dove r è la distanza dal baricentro fra i conduttori [2,12].

Una giustificazione intuitiva di tale comportamento si ottiene ricordando che la legge di Biot – Savart prevede, nel caso un conduttore rettilineo infinito percorso da corrente, un andamento del campo di induzione magnetica inversamente proporzionale alla distanza dal filo.

Per una terna equilibrata, essendo nulla la corrente totale che attraversa una sezione della linea, si perde tale dipendenza del tipo 1/r e, per r>>1, il termine dominante è $1/r^2$.

A ciascuna tipologia di linea si può allora associare un *fattore di forma* k [$\mu T\ m^2\ A^{-1}$] tale che, essendo I [A] la corrente in ciascun conduttore:

$$B \sim k\, I / r^2$$

Fissato un valore di riferimento B_0, con riferimento alla figura, si ottiene:

$$d\,(I; k, B_0) \sim (k\, I / B_0 - h^2)^{1/2}$$

L'approssimazione è tanto migliore quanto più grande risulta d, ovvero quanto più grande è I e piccola la soglia fissata B_0.

Il fattore k è stato determinato, anziché da considerazioni analitiche, da simulazioni eseguite con CMagnetico nelle ipotesi suddette; il valore di corrente I utilizzato nell'effettivo calcolo delle fasce di attenzione è la corrente media dichiarata dagli esercenti per l'anno 2002.

Le fasce di attenzione, rispondendo ad un principio di cautela, sono state valutate ad h=0, in corrispondenza, cioè, del baricentro della distribuzione dei conduttori, nella zona in cui il campo raggiunge la massima intensità; calcolate mediante la formula analitica suscritta, sono state verificate con lo stesso CMagnetico.

L'unico caso in cui tale principio di cautela viene abbandonato si ha per le zone sulle quali i campi prodotti da più linee, sebbene inferiore ciascuno agli 0.2μT, potrebbero sommarsi costruttivamente e causare un superamento di tale soglia; in questi casi, la relazione di fase arbitraria che sussiste in generale fra le linee non consente una stima dell'effettiva intensità del campo. Aree di questo tipo possono in ogni caso essere individuate sul supporto cartografico nell'ambito dell'ultima fase del progetto; qualora ve ne fossero di densamente edificate, si possono prevedere campagne di misura specifiche.

Tipo linea	X_min [m]	X_med [m]	X_max [m]	ΔH_med [m]	ΔH_max [m]	Descrizione	Linee di utilizzo diffuso	K [μT m²/A]
(1)	7.4	-7.4	0	0	1.0	delta; tensione tipica: 380 kV (TERNA)	21356; 21347; 21321; 21361	2.69
(2)	5.25	-4.5	4.0	3.0	6.0	s.t.; tensione tipica: 220kV (TERNA)	22186; 22195; 28566	2.05
(3)	2.8 ÷ 3.85	-2.7 ÷ -3.85	2.6 ÷ 3.15	1.95÷ 2.45	3.9÷ 4.9	s.t; tensione tipica: 132kV (RFI TERNA; ENEL DISTIB.; TERNA)	23731; 20000; 40000; 30000; 23421; 23704; 28424; 28428; 28746; …	1.27
(4)	5.1	0.0	5.1	0.0	0.0	delta: tensione tipica: 132kV (ENEL DISTRIB)	28573; 28708	1.80
(5)	4.7	5.7	4.2	5.4	10.8	d.t.; tensione tipica: 380kV	21347; 21356	3.87
(6)	3.9	3.5	3.1	4.7	9.4	d.t.; tensione tipica: 132kV	28725; 28733	3.28

Tabella 1: tipologie di linea individuate: X_min, X_med, X_max indicano gli sbracci dei conduttori minimo, medio e massimo rispetto all'asse della linea; ΔH_med, ΔH_max indicano le differenze di altezza fra il conduttore minimo e gli altri due; k è il fattore di forma stimato per ciascuna tipologia.

5. Determinazione del numero di edifici interessati

Al 21 ottobre 2003, ENEL Distribuzione S.p.A. aveva risposto alle richieste di ARPA fornendo i dati di corrente relativi all'83% delle linee esercite sul territorio della provincia di Udine, omettendo le seguenti:

28420	Ponterosso	-	Codroipo
28423	Redipuglia	-	Schiavetti
28424	Schiavetti	-	Belvedere
28566	Udine Ovest	-	Udine Rotonda
28733	Redipuglia	-	Ca' Poia
28781	Tarvisio	-	Camporosso FS

Ciò ha impedito la definizione delle fasce di attenzione per tali linee e, conseguentemente, l'individuazione completa degli edifici interessati nei seguenti Comuni:

1. AQUILEIA
2. BASILIANO
3. CAMPOLONGO AL TORRE
4. CERVIGNANO DEL FRIULI
5. CODROIPO
6. FIUMICELLO
7. MARTIGNACCO
8. PASIAN DI PRATO
9. RUDA
10. TARVISIO
11. TORVISCOSA
12. UDINE

I dati riguardanti le rimanenti linee, viceversa, hanno consentito di individuare i seguenti edifici, suddivisi secondo le classi previste dalla Carta Tecnica Regionale Numerica e considerate interessanti ai fini del censimento:

TIPOLOGIA	*NUMERO*
Edificio Civile	606
Edificio Argroforestale	71
Stabilimento Industriale	135
Tettoia	150
TOTALE	***962***

suddivisi nei Comuni interessati come segue:

	COMUNE	*NUMERO*
1.	AIELLO DEL FRIULI	10
2.	AMARO	4
3.	AMPEZZO	10
4.	ARTEGNA	10
5.	BAGNARIA ARSA	5
6.	BUIA	33
7.	CAMPOFORMIDO	42
8.	CARLINO	20
9.	CASTIONS DI STRADA	17
10.	CAVAZZO CARNICO	2
11.	CERCIVENTO	4
12.	CHIOPRIS-VISCONE	2

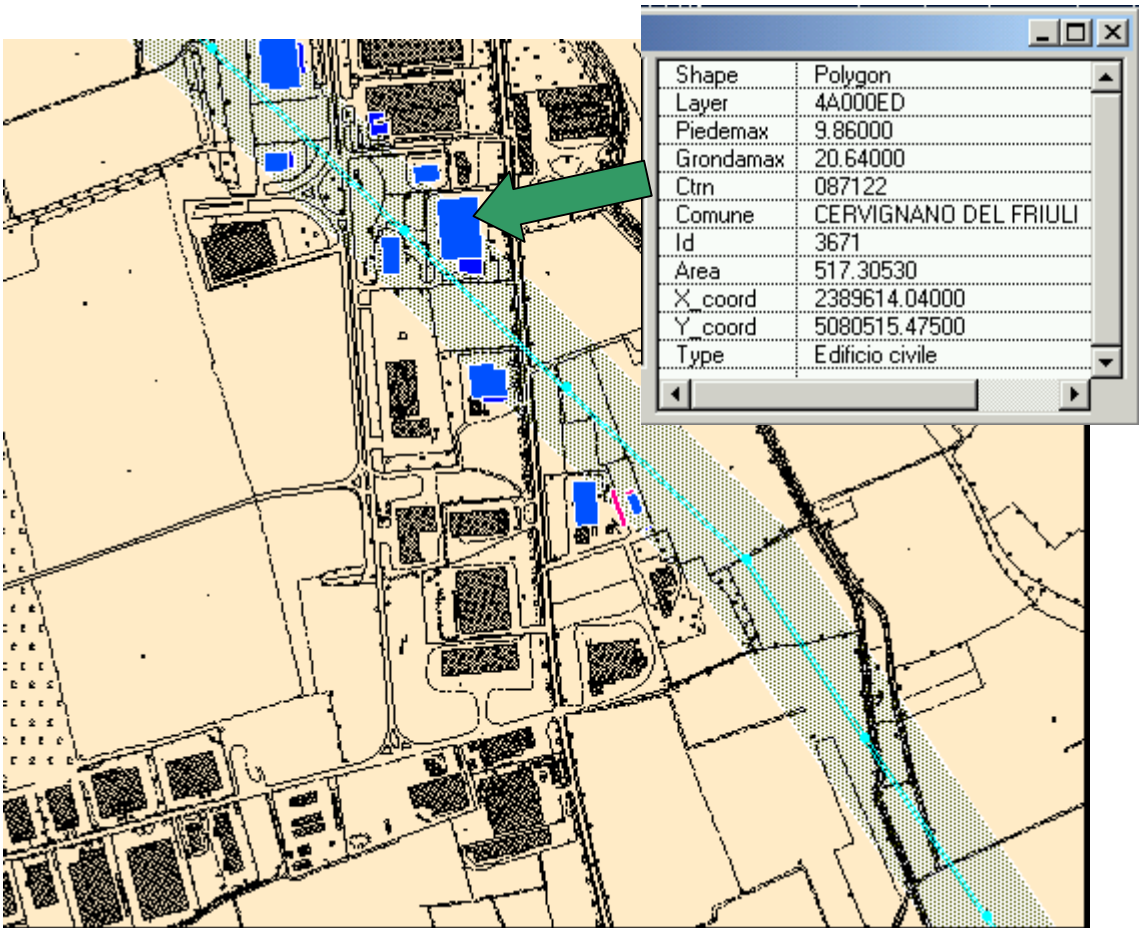


Figura 6: esempio di edifici interessati dalla fascia di attenzione. Fra i dati disponibili, la massima quota di gronda.

13.	CHIUSAFORTE	10
14.	COLLOREDO DI MONTALBANO	12
15.	COSEANO	12
16.	DOGNA	6
17.	ENEMONZO	1
18.	FAGAGNA	20
19.	FLAIBANO	1
20.	FORGARIA NEL FRIULI	32
21.	GEMONA DEL FRIULI	75
22.	GONARS	7
23.	LATISANA	40
24.	LESTIZZA	9
25.	MAGNANO IN RIVIERA	6
26.	MAJANO	1
27.	MALBORGHETTO VALBRUNA	15
28.	MERETO DI TOMBA	11
29.	MOGGIO UDINESE	12
30.	MOIMACCO	8
31.	MORTEGLIANO	4
32.	MORUZZO	3
33.	MUZZANA DEL TURGNANO	1
34.	OSOPPO	10
35.	PAGNACCO	44
36.	PALAZZOLO DELLO STELLA	4
37.	PALMANOVA	11
38.	PAVIA DI UDINE	32
39.	POCENIA	9
40.	PONTEBBA	58
41.	PORPETTO	7
42.	POVOLETTO	13
43.	POZZUOLO DEL FRIULI	8
44.	PARADAMANO	17
45.	PRECENICCO	2
46.	PREMARIACCO	3
47.	PREONE	2
48.	REANA DEL ROIALE	16
49.	REMANZACCO	60
50.	RESIUTTA	7
51.	RIVE D'ARCANO	8
52.	S.DANIELE DEL FRIULI	11
53.	SAN GIORGIO DI NOGARO	58

54.	SAN GIOVANNI AL NATISONE	7
55.	SAN VITO AL TORRE	1
56.	SEDEGLIANO	2
57.	SOCCHIEVE	5
58.	SUTRIO	9
59.	TALMASSONS	3
60.	TARCENTO	8
61.	TAVAGNACCO	16
62.	TOLMEZZO	12
63.	TRASAGHIS	12
64.	TREPPO GRANDE	1
65.	TRICESIMO	4
66.	TRIVIGNANO UDINESE	17
67.	VENZONE	30
68.	VERZEGNIS	7
69.	ZUGLIO	1

7. Bibliografia

1. D.P.C.M. 8.7.2003, *"Fissazione dei limiti di esposizione, dei valori di attenzione e degli obiettivi di qualità per la protezione della popolazione dalle esposizioni ai campi elettrici e magnetici a frequenza di rete (50 Hz) generati dagli elettrodotti"*; art.5, comma 3; G.U. 28.08.2003
2. Norma CEI 211-6, *"Guida per la misura e per la valutazione dei campi elettrici e magnetici nell'intervallo di frequenza 0 Hz - 10 kHz, con riferimento all'esposizione umana"* (2001)
3. L.Anglesio et al., *Guida tecnica per la misura dei campi elettromagnetici compresi nell'intervallo di frequenza 100 kHz – 3 GHz in riferimento all'esposizione della popolazione,* ANPA RTI CTN_AGF 1/2000 (2000)

   http://www.sinanet.anpa.it/documentazione/ CTNAGF/RTI CTN_AGF 1_2000.pdf
4. G.Licitra et al., *Rassegna dei modelli per il rumore, i campi elettromagnetici e la radioattività ambientale,* ANPA, RTI CTN_AGF 1/2001 (2001)

   http://www.sinanet.anpa.it/documentazione/ CTNAGF/RTI CTN_AGF 1_2001.pdf
5. E.Gambato, M.Rosa, "Studio dei modelli per il calcolo dei campi elettromagnetici generati da elettrodotti", ARPAV – Osservatorio Regionale agenti Fisici (documento tecnico, 05/11/00)
6. Delib.G.R. n.1526, 11.4.2000, *L.R. 3.6.1993 n. 27 e successive modificazioni ed integrazioni: "Prevenzione dai campi elettromagnetici generati da elettrodotti. Direttive"*
7. R.Turri, M.Albano, *Calcolo previsionale dei campi elettromagnetici generati da elettrodotti,* AEI – Atti della Giornata di Studio "Elettrodotti e Territorio"; Padova, 21.11.2000 (2001)
8. B.M.Stievano, M.Erna, *Rassegna degli effetti derivanti dall'esposizione a campi elettromagnetici*, ANPA, RTI CTN_AGF 2_2000, 2000

   http://www.sinanet.anpa.it/documentazione/ CTNAGF/RTI CTN_AGF 2_2000.pdf
9. per una descrizione del sistema di misura dei campi elettrici, magnetici ed elettromagnetici EMDEX II: www.enertech.net
10. J. R. Taylor, *Introduzione all'analisi degli errori*, Zanichelli, 1986
11. G.D'Amore et al., *Standard per la realizzazione delle banche dati delle sorgenti di inquinamento elettromagnetico (alte e basse frequenze)*, ANPA, RTI CTN_AGF 4_2001, 2001

http://www.sinanet.anpa.it/documentazione/ CTNAGF/RTI CTN_AGF 4_2001.pdf

12. G.Anderle et al., "IMPATTO AMBIENTALE DA CAMPI ELETTRICI E MAGNETICI A FREQUENZA DI RETE" (documento tecnico nell'ambito del Progetto di ricerca *"Impatto ambientale da Campi Elettrici e Magnetici alla Frequenza di Rete"*, Provincia di Trento e Istituto Trentino di Cultura

    http://www.provincia.tn.it/appa/tecnico/rumore/elf/TestoELF.pdf

13. *754-1985 IEEE Standard for Binary Floating-Point Arithmetic 1985*

APPENDICE A

Il programma CMagnetico, realizzato dall'Istituto Trentino di Cultura per conto dell'Agenzia Provinciale per la Protezione dell'Ambiente di Trento, consente di simulare la mappa tridimensionale dell'induzione magnetica generata da linee AT/AAT, dati alcuni parametri delle campate e dei sostegni considerati.

Come già illustrato, tale applicativo, acquistato da ARPA FVG, è stato integrato nell'ambito di questo Studio con il Catasto degli elettrodotti AT/AAT presenti nella provincia di Udine, in ambiente GIS ArcView 3.2.

A1. Formato dei dati e dei file utilizzati da CMagnetico: *lcm

L'applicativo CMagnetico prevede una schermata di immissione dei parametri della simulazione; i dati inseriti, opportunamente elaborati, vengono salvati in un file con suffisso *.lcm.

Il formato di tale file è stato decodificato nel seguente prospetto:

- **header complessivo:**

from byte	to byte	value	Type	Content
0	7	Scalar	Double Precision (8byte)	Minima dist.dal filo
8	15	Scalar	Double Precision (8byte)	Accuratezza
16	21	Array	Unsigned integer (2byte)	# punti campionamento: [n_y, n_z, n_z]
22	45	Array	Double Precision (8byte)	Passo campionamento [m]: [s_x,s_y,s_z]
46	69	Array	Double Precision (8byte)	Origine campionamento [m]: [x_0, y_0, z_0]
70	71	Scalar	Unsigned Integer (2byte)	Numero di conduttori

- **blocchi di dati**, uno per ogni conduttore, all'interno dei quali c'e' un header, seguito dai blocchi per le campate; l'ultima campata di ogni conduttore presenta anche il valore di H2:

  Es.: un conduttore sviluppato su due campate - in giallo, header dedicato al conduttore; in blu la prima campata; in verde la seconda:

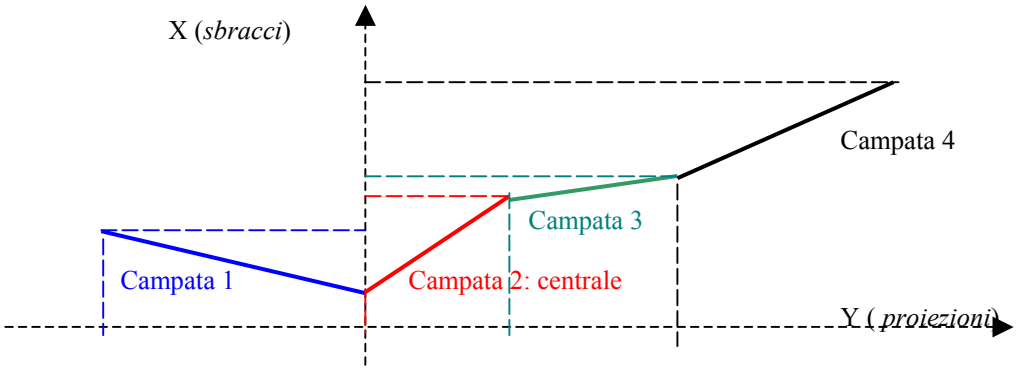
from byte	to byte	value	type	Content
0	1	Scalar	Unsigned Integer (2byte)	Numero di campate
2	9	Scalar	Double Precision (8byte)	Corrente [A]
10	11	Scalar	Unsigned Integer (2byte)	Fase {0;1;2}->{R;S;T}
12	19	Scalar	Double Precision (8byte)	Offset
20	27	Scalar	Double Precision (8byte)	1/Param_catenaria [1/m]
28	35	Scalar	Double Precision (8byte)	Lunghezza campata [m]
36	43	Scalar	Double Precision (8byte)	H1 [m]
44	51	Scalar	Double Precision (8byte)	U0 [m]
52	59	Scalar	Double Precision (8byte)	H0 [m]
60	67	Scalar	Double Precision (8byte)	Sbraccio [m]
68	75	Scalar	Double Precision (8byte)	Proiezione [m]
76	83	Scalar	Double Precision (8byte)	Inclinazione [rad]
84	91	Scalar	Double Precision (8byte)	1/Param_catenaria [1/m]
92	99	Scalar	Double Precision (8byte)	Lunghezza campata [m]
…	…	…	…	…
140	147	Scalar	Double Precision (8byte)	Inclinazione [rad]
148	155	Scalar	Double Precision (8byte)	H2 [m]

Per il significato dei simboli H0, H1, H2, U0, ci si riferisca a quanto descritto nel Glossario dell'Allegato all'elaborato conclusivo della prima parte di questo progetto (prot. ARPA n.10347/2003/TQ/IA/113).

Il *parametro della catenaria* cui si fa riferimento, è il parametro *a* che compare nella seguente espressione della curva catenaria:

$$h = b + a \cosh((x-x0)/a) = b + 1/2\, a \,(\exp (- (x-x0) / a) + \exp ((x-x0) / a))$$

I parametri *sbraccio* e *proiezione* sono indicati nel disegno che segue; secondo le convenzioni di CMagnetico, l'asse y è l'asse principale della linea. L'origine dell'asse y coincide con l'inizio della campata centrale (nel caso del disegno di Figura A1, la seconda). Nel disegno, gli sbracci e le proiezioni da associare a ciascuna campata sono tratteggiati e rappresentati con lo stesso colore della campata corrispondente. L'*inclinazione* è l'angolo formato da ciascuna campata rispetto all'asse y.



I dati sono scritti su file nei formati previsti dallo standard IEEE-754 [13], con la convenzione *little endian*.

Figura A1: schema esemplificativo del sistema di riferimento usato da CMagnetico

A2. Formato dei dati e dei file utilizzati da CMagnetico: .*vcm

Il risultato di una simulazione viene salvato da CMagnetico in un file con suffisso *.vcm. L'organizzazione dei dati è la seguente:

from byte	to byte	value	Type	content
0	3	Scalar	Floating point (4byte)	Minima dist.dal filo
4	5	Scalar	Unsigned integer (2byte)	# punti campionamento x: n_x
6	7	Scalar	Unsigned integer (2byte)	# punti campionamento y: n_y
8	9	Scalar	Unsigned integer (2byte)	# punti campionamento z: n_z
10	13	Scalar	Floating point (4byte)	Passo campionamento x
14	17	Scalar	Floating point (4byte)	Passo campionamento y
18	21	Scalar	Floating point (4byte)	Passo campionamento z
22	25	Scalar	Floating point (4byte)	Origine x
26	29	Scalar	Floating point (4byte)	Origine y
30	33	Scalar	Floating point (4byte)	Origine z
30	33 + 16 n_x n_y n_z	Array	Floating point (4byte)	Dati

I dati sono una matrice di dimensioni: 4 x (n_x) x (n_y) x (n_z), essendo ogni punto caratterizzato dalle 3 componenti ortogonali del campo e dal modulo. Il valore di ciascuna componente è un numero floating-point (4 bytes).

Detto c ~~e~~ {0,1,2,3}, con la convenzione:

c = 0 -> componente x dell'induzione magnetica
c = 1 -> componente y dell'induzione magnetica
c = 2 -> componente z dell'induzione magnetica
c = 3 -> modulo dell'induzione magnetica

i dati sono salvati nel seguente modo:

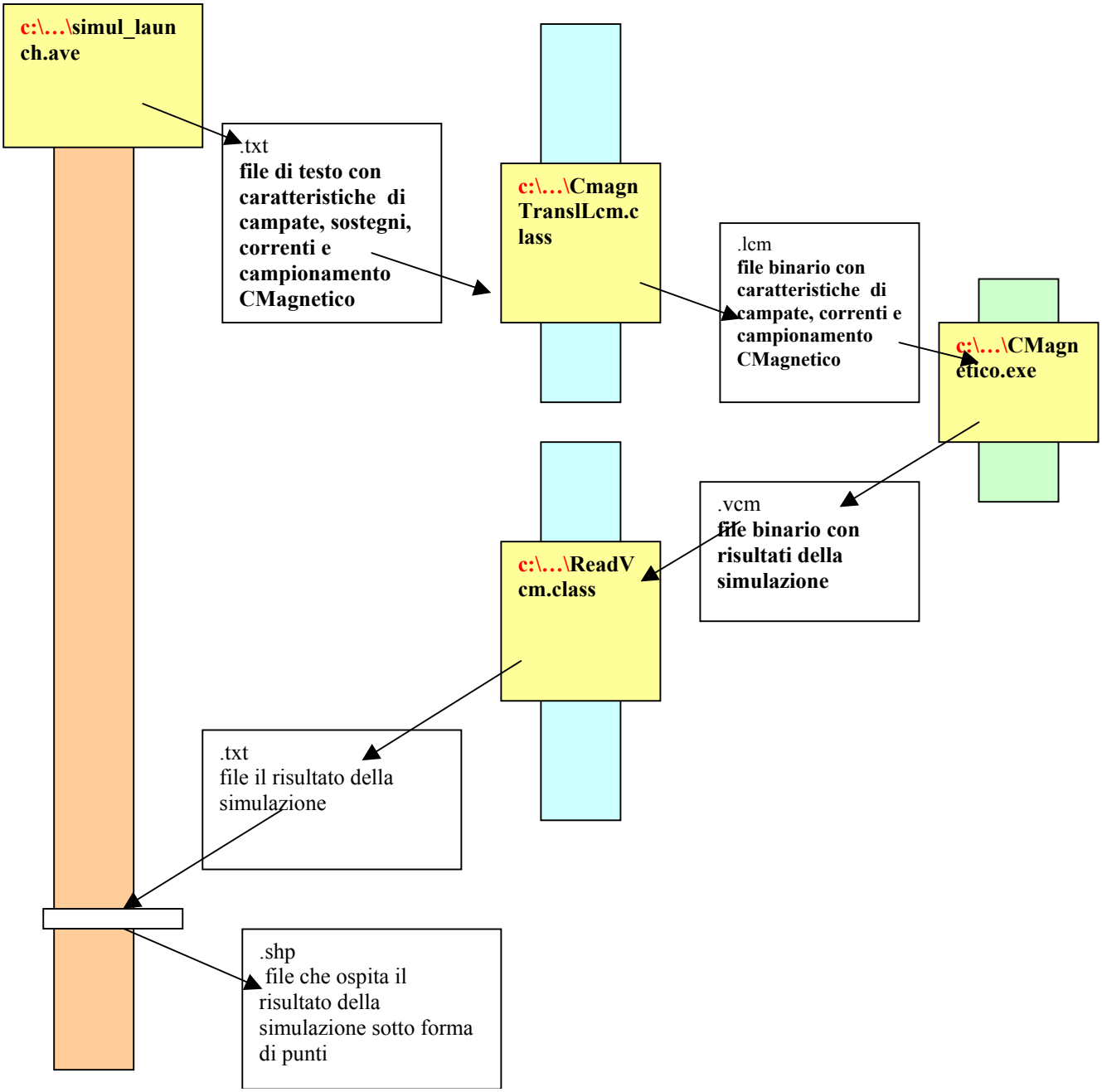


Figura A2: diagramma sul funzionamento dell'integrazione CMagnetico – catasto elettrodotti.

la componente c del campo misurata nella cella (x,y,z) occupa, nel vettore di dati, la posizione:

$$p(c, x, y, z) = 4\, n_x\, n_y\, z + 4\, n_x\, y + 4\, x + c$$

Cioe', si legge a partire dal byte:

$$33 + 4\,(4\, n_x\, n_y\, z + 4\, n_x\, y + 4\, x + c)$$

ed occupa 4 bytes.

A3. Integrazione con il Catasto in GIS ArcView 3.2:

L'integrazione con il catasto è stata realizzata per mezzo di:

- uno script di ArcView 3.2, che utilizza il linguaggio Avenue *(simul_launch.ave)*;

- due programmi JAVA, composti complessivamente da tre classi create opportunamente *(Swab.java; CmagnTranslLcm.java;ReadVcm.java)*;

Data una selezione di campate nel database, lo script Avenue:

- estrae i dati utili dai database delle campate e dei tralicci;
- consente di impostare manualmente alcuni dei parametri della simulazione (numero di punti, passo ed origine del campionamento);
- salva tali dati e parametri in un file di testo;
- lancia il programma JAVA che traduce il file di testo salvato in un file *.lcm per l'utilizzo con CMagnetico;
- lancia il programma JAVA che traduce in file di testo il file *.vcm prodotto da CMagnetico;
- carica i dati della simulazione in un opportuno shapefile *.shp;

I listati dello *script* di Avenue e dei programmi JAVA sono riportati al capo A4.

Il ricorso a JAVA è dovuto al fatto che Avenue consente solamente la scrittura di file di testo (ASCII), e non la scrittura di file in cui i dati si presentino in forma binaria, come i file *.lcm e *.vcm.

Va osservato che, nella scrittura su file, JAVA, diversamente da CMagnetico, adotta la convenzione *big endian*, richiedendo l'invesione dell'ordine dei byte, operata dalla classe *Swab.class*, definita allo scopo.

A4. Codici utilizzati

script Avenue *simul_launch.ave*

```
''05/09/03
''nel database elettrodotti:
''- seleziona e salva i dati per simulazione CMagnetico;
''- carica i risultati della simulazione in uno shapefile opportuno;
''========================= Per le campate selezionate,
''========================= salva il file di testo da tradurre in *.lcm per uso CMagnetico
''evita la notazione esponenziale:
   Number.SetDefFormat( "ddddddd.ddd" )
''VTabs in questione
   tableCamp = av.FindDoc("Attributes of Campate.shp")
   tableSost = av.FindDoc("Attributes of Sostegni.shp")
'   vTabCamp = tableCamp.GetVTab
   vTabSost = tableSost.GetVTab
''campi coinvolti nelle due tabelle
   fieldPoint = vTabCamp.FindField("Shape")
   fieldSost  = vTabSost.FindField("Numpic")
   fieldX     = vTabSost.FindField("X_coord")
   fieldY     = vTabSost.FindField("Y_coord")
   fieldQ     = vTabSost.FindField("Elevation")
   fieldSMin  = vTabSost.FindField("S_min")
   fieldSMed  = vTabSost.FindField("S_med")
   fieldSMax  = vTabSost.FindField("S_max")
   fieldCatena= vTabSost.FindField("Catena")
   fieldHMin  = vTabSost.FindField("H_min")
```

```
  fieldHMax  = vTabSost.FindField("H_max")
  fieldLinea = vTabSost.FindField("Codlinea")
  fieldT_p   = vTabCamp.FindField("T_p")
  fieldPrec  = vTabCamp.FindField("Pic_prec")
  fieldSuc  = vTabCamp.FindField("Pic_suc")
  fieldLung= vTabCamp.FindField("Lunghezza")
  fieldCamp  = vTabCamp.FindField("Id_camp")
  fieldShape = vTabCamp.FindField("Shape")
''bitmap dei sostegni
  sostBitmap = vTabsost.GetSelection
''trova la selezione delle campate...
  campBitmap = vTabCamp.GetSelection
''quante sono...
  numcamp=vtabcamp.GetNumSelRecords
''campata centrale: da 0 a numcamp-1...
  ncentrale=(((numcamp)/2).ceiling)-1
''arrotonda eventualmente all'intero inferiore
  ncentrale.setformatprecision(0)
''crea gli arrays che servono:
''nome campata
  campata={}
''lunghezza campata
  lung={}
''catenaria
  Tc={}
''coordinate
  x0={}
  x1={}
  y0={}
  y1={}
''altezze
  h1min={}
  h1med={}
  h1max={}
  h1minsuc={}
  h1medsuc={}
  h1maxsuc={}
''minimo catenaria
  u0={}
  L={}
''inclinazione campata
```

```
   incl={}
''coseno
   proj={}
''seno
   proj1={}
''sbracci calcolati (per CMagnetico)
   smin={}
   smed={}
   smax={}
''punti di inizio
   inipoint={}
''punti di fine
   finpoint={}
 '*****************************************************
''corre sulle campate selezionate
''indice delle campate, incrementato ad ogni ciclo
view1= av.finddoc("view1")
disp=view1.GetDisplay
   n=0
   n.setformat("d")
''ciclo sulle campate selezonate
   for each camprec in campBitmap
   'nome campata
      campata.add(vtabCamp.ReturnValueString(fieldCamp,camprec))
   'lunghezza campata:
      length=vtabCamp.ReturnValueNumber(fieldLung,camprec)
      lung.add(length)
   'catenaria
      Tc.add(vtabCamp.ReturnValueNumber(fieldT_p,camprec))
   'ricava il nome della linea e dei picchetti per la query nei sostegni
       linea    = vtabCamp.ReturnValueString(fieldCamp,camprec).left(5)
       picprec = vtabCamp.ReturnValueString(fieldPrec,camprec)
       picsuc = vtabCamp.ReturnValueString(fieldSuc,camprec)
   'sostegno precedente
       theQuery = "([Codlinea] = " + linea.quote + ") AND ("+ "[Numpic] = " + picprec.quote+")"
       vtabsost.Query(theQuery, sostBitmap, #VTAB_SELTYPE_NEW)
   'scarica i dati del sostegno
        sostprec=sostBitmap.getnextset(-1)
        quota=vtabSost.ReturnValueNumber(fieldQ,sostprec)
        isolatori=vtabSost.ReturnValueNumber(fieldCatena,sostprec)
   'definisce le quote d'attacco dei fili (= quota slm + altezza mensole - isolatori)
```

```
    h1min.add(quota+(vtabSost.ReturnValueNumber(fieldHMin,sostprec))-(isolatori))
    h1max.add(quota+(vtabSost.ReturnValueNumber(fieldHMax,sostprec))-(isolatori))
'per la quota della fune media, definisce se il sostegno è a delta o no...
    if (vtabSost.ReturnValueNumber(fieldSMax,sostprec)=0) then
      h1med.add(quota+vtabSost.ReturnValueNumber(fieldHMin,sostprec)-(isolatori))
    else
      h1med.add(((2*quota+vtabSost.ReturnValueNumber(fieldHMin,sostprec)
      +vtabSost.ReturnValueNumber(fieldHMax,sostprec))/2)-(isolatori))
    end
'sostegno seguente
   theQuery = "([Codlinea] = " + linea.quote + ") AND ("+ "[Numpic] = " + picsuc.quote+")"
   vtabsost.Query(theQuery, sostBitmap, #VTAB_SELTYPE_NEW)
'scarica i dati del sostegno seguente
    sostsuc=sostBitmap.getnextset(-1)
    quotasuc=vtabSost.ReturnValueNumber(fieldQ,sostsuc)
    isolatori=vtabSost.ReturnValueNumber(fieldCatena,sostsuc)
    h1minsuc.add(quotasuc+(vtabSost.ReturnValueNumber(fieldHMin,sostsuc))-(isolatori))
    h1maxsuc.add(quotasuc+(vtabSost.ReturnValueNumber(fieldHMax,sostsuc))-(isolatori))
'sost a delta o no...
    if (vtabSost.ReturnValueNumber(fieldSMax,sostsuc)=0) then
      h1medsuc.add(quotasuc+vtabSost.ReturnValueNumber(fieldHMin,sostsuc)-(isolatori))
    else
      h1medsuc.add(((2*quotasuc+vtabSost.ReturnValueNumber(fieldHMin,sostsuc)
      +vtabSost.ReturnValueNumber(fieldHMax,sostsuc))/2)-(isolatori))
    end
'mette i punti in lista
   thisinipoint=vtabSost.ReturnValue(fieldPoint,sostprec)
   thisfinpoint=vtabSost.ReturnValue(fieldPoint,sostsuc)
   inipoint.add(thisinipoint)
   finpoint.add(thisfinpoint)
    disp.drawtext(inipoint.get(n),n.asstring,symbol.make(#SYMBOL_TEXT ))
'*********
'nel caso della campata centrale
'sbracci della campata centrale: XMIN,XMED,XMAX
    if (n=ncentrale) then
    xcoordcentr=inipoint.get(n).getx
    ycoordcentr=inipoint.get(n).gety
    x1coordcentr=finpoint.get(n).getx
    y1coordcentr=finpoint.get(n).gety
    sinus=(y1coordcentr-ycoordcentr)/length
    cosinus=(x1coordcentr-xcoordcentr)/length
```

```
        anAngle=(sinus/cosinus).atan
    'estende il range angolare da - PI a + PI
        if (cosinus>0) then
            angle0=anangle
        else
            if (sinus>0) then
                angle0=anAngle+3.14159
            else
                angle0=anAngle-3.14159
            end
        end
          'angolo della campata rispetto l'asse y
          myanglebox=(angle0-1.57080)
          msgbox.report("angle:"+myanglebox.asdegrees.asstring+" ;
           xc:"+xcoordcentr.asstring+" ; yc:"+ycoordcentr.asstring,"trasformazione:")
          xmin=vtabSost.ReturnValueNumber(fieldSMin,sostprec)
    'ATTENZIONE AL SEGNO!!!!
          xmed=-(vtabSost.ReturnValueNumber(fieldSMed,sostprec))
          xmax=vtabSost.ReturnValueNumber(fieldSMax,sostprec)
        end
    '*******
    'incrementa il numero di campata
        n=n+1
        n.setformat("d")
end
''creo una trasformazione per il sist.rif. della campata centrale
anewTrans=transform2d.make
''grossa traslazione
anewTrans.move((-xcoordcentr)@(-ycoordcentr))
''ruoto: attenzione!
''l'asse y del campionamento deve essere portato a coinciere con l'asse della campata
anewTrans.rotate(-(myanglebox.asdegrees))
''prepara le proiezioni delle campate
for each p in 0..(numcamp-1)
''rototrasla i punti di inizio e fine campate
inipoint.get(p).transform(anewtrans)
finpoint.get(p).transform(anewtrans)
''le coordinate
xx=inipoint.get(p).getx
yy=inipoint.get(p).gety
xx1=finpoint.get(p).getx
```

```
yy1=finpoint.get(p).gety
''scrive l'inclinazione
incl.add((((xx1-xx)/(yy1-yy))).atan)
''tiene le nuove coordinate
proj1.add(xx)
proj.add(yy)
end
'====================SCRITTURA DEL FILE==================
''definisce la directory iniziale
   mypath="c:\"
''prepara il file lf
   result=FileDialog.Put (mypath.asfilename, "*.txt", "Extract parameters to:")
   lf = LineFile.Make( result, #FILE_PERM_WRITE )
   lf.WriteElt(Date.Now.asstring)
   lf.WriteElt("Data saved from Catasto Elettrodotti della Provincia di Udine")
   lf.WriteElt("in order to be used with the CMagnetico code:")
''scrive nel file
  ncentrale.setformat("d")
  lf.WriteElt("Numero campate:")
  lf.WriteElt(numcamp.setformat("d").asstring)
  lf.WriteElt("La campata centrale è la:")
  lf.WriteElt(ncentrale.asstring)
''indice nelle liste
   m=0
for each camprec in campBitmap
  lf.WriteElt("***************")
  'butta i decimali
  m.setformat("d")
  'numero campata:
  lf.WriteElt("N° campata: "+m.asstring)
  'Id Campata
  lf.WriteElt("Id campata: "+campata.get(m).asstring)
  'Catenaria
  lf.WriteElt("Catenaria [1/m]:")
  lf.WriteElt((1/(tc.get(m))).setformat("d.dddddd").asstring)
  'Lunghezza
  lf.WriteElt("Lunghezza [m]:")
  lf.WriteElt(lung.get(m).asstring)
  'Coordinata
  lf.WriteElt("Coordinata [m]:")
  lf.WriteElt(proj.get(m).asstring)
```

```
'Inclinazione
 lungnumb=lung.get(m)
 lf.WriteElt("Inclinazione [rad]:")
 lf.writeElt(incl.get(m).asstring)
'Sbracci
lf.WriteElt("Sbraccio fune minima [m]:")
lf.writeelt((xmin+proj1.get(m)).asstring)
lf.WriteElt("Sbraccio fune media [m]:")
lf.writeelt((xmed+proj1.get(m)).asstring)
lf.WriteElt("Sbraccio fune massima [m]:")
lf.writeelt((xmax+proj1.get(m)).asstring)
lf.WriteElt("H1 fune minima [m]:")
lf.writeElt(H1min.get(m).asstring)
lf.WriteElt("H1 fune media:")
lf.writeElt(H1med.get(m).asstring)
lf.WriteElt("H1 fune massima:")
lf.writeElt(H1max.get(m).asstring)
'dal T/p, H1, H2 ricava U0
Tcm=Tc.get(m)
lungnum=lung.get(m)
lf.WriteElt("U_0 [m]:")
U0=(((h1minsuc.get(m)-h1min.get(m))*Tcm*2)-(lungnum^2))/(-2*lungnum)
lf.writeelt(u0.asstring)
'ricava H0min.med.max
lf.WriteElt("H_0 min [m]:")
H0min=h1min.get(m)-((u0^2)/(2*tcm))
lf.writeelt(h0min.asstring)
lf.WriteElt("H_0 med [m]:")
H0med=h1med.get(m)-((u0^2)/(2*tcm))
lf.writeelt(h0med.asstring)
lf.WriteElt("H_0 max [m]:")
H0max=h1max.get(m)-((u0^2)/(2*tcm))
lf.writeelt(h0max.asstring)
"scrive h2 per l'ultima campata
if (m=(numcamp-1)) then
 lf.WriteElt("H_2 fune minima [m]:")
 lf.writeelt(h1minsuc.get(m).asstring)
 lf.WriteElt("H_2 fune media [m]:")
 lf.writeelt(h1medsuc.get(m).asstring)
 lf.WriteElt("H_2 fune massima [m]:")
 lf.writeelt(h1maxsuc.get(m).asstring)
```

```
  end
m=m+1
m.setformat("d")
end
''altre info:
''chiedo i parametri del campionamento (n°di punti; step) nelle tre direzioni
labels = { "# of points along x:", "# of points along y:","# of points along z:",
        "x sampling distance:","y sampling distance:","z sampling distance:","x origin:",
        "y origin:","z origin:","CURRENT:" }
defaults = { "200", "200", "3","10","10","5","-1000","-1000","10","1000" }
realvalues = MsgBox.MultiInput( "Can you please indicate these simulation settings?", "Realvalues", labels,
defaults )
for each ndata in 0..9
lf.writeelt(labels.get(ndata))
lf.writeelt(realvalues.get(ndata))
end
''per la visualizzazione successiva sul GIS dei risultati della simulazione:
lf.writeelt("Coordinate dell'origine sist.rif. CMagnetico:")
lf.writeelt(xcoordcentr.asstring)
lf.writeelt(ycoordcentr.asstring)
lf.writeelt("Punto finale della campata centrale:")
lf.writeelt(x1coordcentr.asstring)
lf.writeelt(y1coordcentr.asstring)
lf.Close
''lancia il JAVA...
System.Execute("c:\j2sdk1.4.0\bin\java.exe -cp c:\corso_java\cmagnetico_java CMagnTranslLcm "
            +(result.asstring))
''file .vcm e file di testo per ArcView
resultvcm=filedialog.show("*.vcm","*.vcm","Open a .vcm volume:")
resultvolume=(resultvcm.asstring.substitute(".vcm","1.txt")).asfilename
stringone="c:\j2sdk1.4.0\bin\java.exe -cp c:\corso_java\cmagnetico_java ReadVcm "
            +(resultvcm.asstring)
System.Execute(stringone)
'=================================================================================
'================== Prepara un file .shp di tipo POINT per la successiva visualizzazione dei dati
'=================================================================================
''dati della campata centrale e parametri della simulazione
lenght=(((x1coordcentr-xcoordcentr)^2)+((y1coordcentr-ycoordcentr)^2)).sqrt
xnum=realvalues.get(0).asnumber
ynum=realvalues.get(1).asnumber
znum=realvalues.get(2).asnumber
```

```
xstep=realvalues.get(3).asnumber
ystep=realvalues.get(4).asnumber
zstep=realvalues.get(5).asnumber
xorigin=realvalues.get(6).asnumber
yorigin=realvalues.get(7).asnumber
zorigin=realvalues.get(8).asnumber
''creo una trasformazione per il posizionamento degli shapepoints
aTrans=transform2d.make
''piccolo shift
aTrans.move(xorigin@yorigin)
''ruoto: attenzione!
''l'asse y del campionamento deve essere portato a coinciere con l'asse della campata
aTrans.rotate(myanglebox.asdegrees)
''grossa traslazione
aTrans.move(xcoordcentr@ycoordcentr)
''becco il display attivo
 view1= av.finddoc("view1")
 disp=view1.GetDisplay
''costruisco, come tema, una griglia di POINTS sovrapposta al poligono
''file in cui riporre i dati
 mainFile=FileDialog.Put ("c:\".asfilename,"*.shp", "Select the shp file for writing:")
''nuova FTab, per il file con i punti quotati
  mainFTab=FTab.MakeNew( mainFile, POINT )
  mainpointField=mainFTab.FindField("Shape")
''campi della nuova FTab
  mainidfield=Field.Make("Id", #FIELD_CHAR, 9, 0)
  mainfieldx=Field.Make("X_Coord", #FIELD_DECIMAL, 16, 5)
  mainfieldy=Field.Make("Y_Coord", #FIELD_DECIMAL, 16, 5)
  mainnote=Field.Make("Note", #FIELD_CHAR, 256, 0)
  mainFTab.addfields({mainidfield,mainfieldx,mainfieldy,mainnote})
''tanti campi quanti gli z samples:
mainb={}
for each numquotas in 0..(znum-1)
  effquotas=zorigin+(numquotas*zstep)
  effquotas.setformat("d.dd")
  mainb.add(Field.Make("|B|["+effquotas.asstring+"]", #FIELD_DECIMAL, 16, 5))
end
''aggiunge la lista di campi
mainFTab.addfields(mainb)
''aggiungo effettivamente i punti
''genera i punti della griglia nel rettangolo
```

```
mainindex=0
for each m in 0..(xnum-1)
  for each n in 0..(ynum-1)
  point_mn=point.make(m*xstep,n*ystep)
  mypoint=point_mn.transform(atrans)
  mainFTab.addrecord
  mainFTab.setvalue(mainpointfield,mainindex,mypoint)
  mainindex=mainindex+1
  end
end
''aggiunge il tema alla vista
myView=av.finddoc("View1")
mainTheme=FTheme.make(mainFTab)
myview.addtheme(maintheme)
''CARICA I DATI DI SIMULAZIONE
''apre un file contenente il volume e carica i dati che trova in un shpfile con points...
lf = LineFile.make( resultvolume, #FILE_PERM_READ )
mainFTab.SetEditable(true)
mainFTab.setvalue(mainnote,0,date.now.asstring)
mainFTab.setvalue(mainnote,1,"Data extracted from:")
mainFTab.setvalue(mainnote,2,result.asstring)
mainFTab.setvalue(mainnote,3,"Current: "+realvalues.get(9)+" A")
  for each nnb in 0..(znum-1)
    nnindex=0
    for each nnx in 0..(xnum-1)
      for each nny in 0..(ynum-1)
        indmagn=lf.readelt.asnumber
        mainFTab.setvalue(mainb.get(nnb),nnindex,indmagn)
        mainFTab.setvalue(mainfieldx,nnindex,mainftab.returnvalue(mainpointfield,nnindex).getx)
        mainFTab.setvalue(mainfieldy,nnindex,mainftab.returnvalue(mainpointfield,nnindex).gety)
        nnindex=nnindex+1
      end
    end
  end
lf.close
'============================
```

classi Java Swab.java; CmagnTranslLcm.java;ReadVcm.java

```
//==================================
//Palmanova, 01-09-03
// classe CmagnTransLcm :
```

```
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.net.*;
import java.nio.*;
public class CMagnTransILcm
{  public static void main(String[] args) throws Exception
   {
   //args[0] deve essere la stringa nome dell'input file (.txt)
     String theInFileNameString = args[0];
   //crea l'output file, togliendo il suffisso .txt e mettendo .lcm
     int theInFileNameStringLength= theInFileNameString.length();
     String theOutFileNameString = theInFileNameString.substring(0,theInFileNameStringLength-3)+"lcm";
   //crea il BufferedReader per l'input:
     File anAViewFile = new File(theInFileNameString);
     FileInputStream aviewFileInputstream = new FileInputStream(anAViewFile);
     InputStreamReader aviewInputStreamReader = new InputStreamReader(aviewFileInputstream);
     BufferedReader aviewFileInput = new BufferedReader(aviewInputStreamReader);
   //crea il file per l'output e gli oggetti necessari alla scrittura
     File theLcmFile = new File(theOutFileNameString);
     theLcmFile.createNewFile();
     FileOutputStream theLcmOutStream = new FileOutputStream(theLcmFile);
     DataOutputStream theLcmDataOutStream= new DataOutputStream(theLcmOutStream);
   //legge 3 righe stupide
     readDumb(aviewFileInput,theLcmDataOutStream);
     readDumb(aviewFileInput,theLcmDataOutStream);
     readDumb(aviewFileInput,theLcmDataOutStream);
   //carica i valori
     readDumb(aviewFileInput,theLcmDataOutStream);
     Integer numCampInt=Integer.valueOf(aviewFileInput.readLine());
     readDumb(aviewFileInput,theLcmDataOutStream);
     Integer centraleInt=Integer.valueOf(aviewFileInput.readLine());
     int numcamp=numCampInt.intValue();
     int centrale=centraleInt.intValue();
   //definisce gli arrays
     long[] catenaria = new long[numcamp];
     long[] lung = new long[numcamp];
     long[] incl = new long[numcamp];
     long[] proj = new long[numcamp];
     long[] sb_min = new long[numcamp];
```

```
long[] sb_med = new long[numcamp];
long[] sb_max = new long[numcamp];
long[] h1_min = new long[numcamp];
long[] h1_med = new long[numcamp];
long[] h1_max = new long[numcamp];
long[] h0_min = new long[numcamp];
long[] h0_med = new long[numcamp];
long[] h0_max = new long[numcamp];
long[] u0     = new long[numcamp];
for (int n = 0;n<(numcamp);n++)
{
  for (int m = 0; m<3; m++)
    {
    readDumb(aviewFileInput,theLcmDataOutStream);
    }
 catenaria[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 lung[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 proj[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 incl[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 sb_min[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 sb_med[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 sb_max[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 h1_min[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 h1_med[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 h1_max[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 u0[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 h0_min[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 h0_med[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 h0_max[n] = translDouble(aviewFileInput,theLcmDataOutStream);
 }
long h2_min=translDouble(aviewFileInput,theLcmDataOutStream);
long h2_med=translDouble(aviewFileInput,theLcmDataOutStream);
long h2_max=translDouble(aviewFileInput,theLcmDataOutStream);
short nx=translShort(aviewFileInput,theLcmDataOutStream);
short ny=translShort(aviewFileInput,theLcmDataOutStream);
short nz=translShort(aviewFileInput,theLcmDataOutStream);
long stepx=translDouble(aviewFileInput,theLcmDataOutStream);
long stepy=translDouble(aviewFileInput,theLcmDataOutStream);
long stepz=translDouble(aviewFileInput,theLcmDataOutStream);
long origx=translDouble(aviewFileInput,theLcmDataOutStream);
long origy=translDouble(aviewFileInput,theLcmDataOutStream);
```

```
    long origz=translDouble(aviewFileInput,theLcmDataOutStream);
    long current=translDouble(aviewFileInput,theLcmDataOutStream);
 //SCRITTURA:
//HEADER :
    double precDouble=0.01 ;
    double accDouble=0.001;
    long precLong=Swab.swabDouble(precDouble);
    long accLong=Swab.swabDouble(accDouble);
    theLcmDataOutStream.writeLong(precLong);
    theLcmDataOutStream.writeLong(accLong);
    theLcmDataOutStream.writeShort(nx);
    theLcmDataOutStream.writeShort(ny);
    theLcmDataOutStream.writeShort(nz);
    theLcmDataOutStream.writeLong(stepx);
    theLcmDataOutStream.writeLong(stepy);
    theLcmDataOutStream.writeLong(stepz);
    theLcmDataOutStream.writeLong(origx);
    theLcmDataOutStream.writeLong(origy);
    theLcmDataOutStream.writeLong(origz);
    short numfili=3;
    theLcmDataOutStream.writeShort(Swab.swabShort(numfili));
//FILO 1
    short ss=numCampInt.shortValue();
    theLcmDataOutStream.writeShort(Swab.swabShort(ss));
    theLcmDataOutStream.writeLong(current);
    short fase=0;
    theLcmDataOutStream.writeShort(Swab.swabShort(fase));
    double offset=0;
    long prova =Swab.swabDouble(offset);
    Long pProva=new Long(prova);
    theLcmDataOutStream.writeLong(Swab.swabDouble(offset));
    for (int c = 0; c<numcamp; c++)
      {
      theLcmDataOutStream.writeLong(catenaria[c]);
      theLcmDataOutStream.writeLong(lung[c]);
      theLcmDataOutStream.writeLong(h1_min[c]);
      theLcmDataOutStream.writeLong(u0[c]);
      theLcmDataOutStream.writeLong(h0_min[c]);
      theLcmDataOutStream.writeLong(sb_min[c]);
      theLcmDataOutStream.writeLong(proj[c]);
      theLcmDataOutStream.writeLong(incl[c]);
```

```
    }
    theLcmDataOutStream.writeLong(h2_min);
//FILO 2
  theLcmDataOutStream.writeShort(Swab.swabShort(numCampInt.shortValue()));
  theLcmDataOutStream.writeLong(current);
  short fase2=1;
  theLcmDataOutStream.writeShort(Swab.swabShort(fase2));
  double offset2=0;
  theLcmDataOutStream.writeLong(Swab.swabDouble(offset2));
  for (int c = 0; c<numcamp; c++)
    {
    theLcmDataOutStream.writeLong(catenaria[c]);
    theLcmDataOutStream.writeLong(lung[c]);
    theLcmDataOutStream.writeLong(h1_med[c]);
    theLcmDataOutStream.writeLong(u0[c]);
    theLcmDataOutStream.writeLong(h0_med[c]);
    theLcmDataOutStream.writeLong(sb_med[c]);
    theLcmDataOutStream.writeLong(proj[c]);
    theLcmDataOutStream.writeLong(incl[c]);
    }
    theLcmDataOutStream.writeLong(h2_med);
//FILO 3
  theLcmDataOutStream.writeShort(Swab.swabShort(numCampInt.shortValue()));
  theLcmDataOutStream.writeLong(current);
  short fase3=2;
  theLcmDataOutStream.writeShort(Swab.swabShort(fase3));
  double offset3=0;
  theLcmDataOutStream.writeLong(Swab.swabDouble(offset3));
  for (int c = 0; c<numcamp; c++)
    {
    theLcmDataOutStream.writeLong(catenaria[c]);
    theLcmDataOutStream.writeLong(lung[c]);
    theLcmDataOutStream.writeLong(h1_max[c]);
    theLcmDataOutStream.writeLong(u0[c]);
    theLcmDataOutStream.writeLong(h0_max[c]);
    theLcmDataOutStream.writeLong(sb_max[c]);
    theLcmDataOutStream.writeLong(proj[c]);
    theLcmDataOutStream.writeLong(incl[c]);
    }
    theLcmDataOutStream.writeLong(h2_max);
  for (int cc = 0; cc<80; cc++)
```

```
        {
          theLcmDataOutStream.writeByte(32);
        }
    theLcmDataOutStream.close();
    aviewFileInputstream.close();
    }
//METHOD: reads a dumb line, then a double value and returns it as a
//       long value, ready to be written in the lcm file
public static long transIDouble(BufferedReader in, DataOutputStream out) throws Exception
      {String inputLine0 = in.readLine();
       String inputLine1 = in.readLine();
       Double doubleTrouble=Double.valueOf(inputLine1);
       double d=doubleTrouble.doubleValue();
       long swbd=Swab.swabDouble(d);
       return swbd;
      }
//METHOD: reads a dumb line, then a long value and writes it to the lcm file
public static long transILong(BufferedReader in, DataOutputStream out) throws Exception
      {String inputLine0 = in.readLine();
       String inputLine1 = in.readLine();
       Long longTrouble = Long.valueOf(inputLine1);
       long d = longTrouble.longValue();
       long swbd=Swab.swabLong(d);
       return swbd;
      }
//METHOD: reads a dumb line, then a long value and writes it to the lcm file
public static short transIShort(BufferedReader in, DataOutputStream out) throws Exception
      {String inputLine0 = in.readLine();
       String inputLine1 = in.readLine();
       Short shortTrouble = Short.valueOf(inputLine1);
       short d = shortTrouble.shortValue();
       short swbd=Swab.swabShort(d);
       return swbd;
      }
//METHOD: reads a dumb line, then a float value and writes it to the lcm file
public static int transIFloat(BufferedReader in, DataOutputStream out) throws Exception
      {String inputLine0 = in.readLine();
       String inputLine1 = in.readLine();
       Float floatTrouble= Float.valueOf(inputLine1);
       float d=floatTrouble.floatValue();
       int swbd=Swab.swabFloat(d);
```

```
      return swbd;
     }
//METHOD: reads a dumb line, then an int value and writes it to the lcm file
public static int transIInt(BufferedReader in, DataOutputStream out) throws Exception
     {String inputLine0 = in.readLine();
      String inputLine1 = in.readLine();
      Float floatTrouble= Float.valueOf(inputLine1);
      int d=floatTrouble.intValue();
      int swbd=Swab.swabInt(d);
      return swbd;
     }
//METHOD: reads a dumb line
public static void readDumb(BufferedReader in, DataOutputStream out) throws Exception
     {
       String inputLine0 = in.readLine();
     }
}
//=======================
//SWAB
//Palmanova, 01-09-03
//converts from BIG_ENDIAN to LITTLE_ENDIAN and vice versa
public class Swab {
//arg: int (32bit = 4bytes); returns: int
  public final static int swabInt(int v) {
    return (v >>> 24) | (v << 24) |
      ((v << 8) & 0x00FF0000) | ((v >> 8) & 0x0000FF00);
    }
//arg: short (16bit = 2bytes) returns: short (16bit)
  public final static short swabShort(short s) {
     int i = (s >>> 8) | (s << 8);
     Integer iTrouble=new Integer (i);
     short sf = iTrouble.shortValue();
     return sf;
  }
//arg: double (64bit = 8bytes) returns: long (32bit)
  public final static long swabDouble(double d) {
    long v = Double.doubleToLongBits(d);
    return (v >>> 56) | (v << 56)
    | ((v << 40) & 0x00FF000000000000L) | ((v >> 40) & 0x000000000000FF00L)
    | ((v << 24) & 0x0000FF0000000000L) | ((v >> 24) & 0x0000000000FF0000L)
    | ((v << 8)  & 0x000000FF00000000L) | ((v >> 8)  & 0x00000000FF000000L);
```

```
    }
//arg: float (32bit = 4bytes) returns: int (32bit)
  public final static int swabFloat(float f) {
    int v = Float.floatToIntBits(f);
    return (v >>> 24) | (v << 24) |
      ((v << 8) & 0x00FF0000) | ((v >> 8) & 0x0000FF00);
    }
//arg: long (64bit = 8bytes) returns: long (32bit)
  public final static long swabLong(long v) {
    return (v >>> 56) | (v << 56)
    | ((v << 40) & 0x00FF000000000000L) | ((v >> 40) & 0x000000000000FF00L)
    | ((v << 24) & 0x0000FF0000000000L) | ((v >> 24) & 0x0000000000FF0000L)
    | ((v << 8)  & 0x000000FF00000000L) | ((v >> 8)  & 0x00000000FF000000L);
    }
//main
  public static void main(String argv[]) {
    }
 }
//=======================
//READVCM
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.net.*;
import java.nio.*;
public class ReadVcm
{
  public static float readFloat(DataInputStream in, PrintStream out) throws Exception
  {
    int anint=0;
    anint=in.readInt();
    int aSwabInt=Swab.swabInt(anint);
    float afloat=Float.intBitsToFloat(aSwabInt);
    return afloat;
    }
  public static short readShort(DataInputStream in, PrintStream out) throws Exception
  {
    short ashort=0;
    ashort=in.readShort();
    short aSwabShort=Swab.swabShort(ashort);
```

```
  System.out.println(aSwabShort);
  return aSwabShort;
  }
public static void transIFloat(DataInputStream in, PrintStream out) throws Exception
 {
  int anint=0;
  anint=in.readInt();
  int aSwabInt=Swab.swabInt(anint);
  float afloat=Float.intBitsToFloat(aSwabInt);
  out.println(afloat);
  }
public static void main(String[] args) throws Exception
 {
 //args[0] deve essere la stringa nome dell'input file (.vcm)
  String theInFileNameString = args[0];
 //crea l'input file
  File aVCMFile = new File(theInFileNameString);
  FileInputStream aVCMInputStream = new FileInputStream(aVCMFile);
  DataInputStream aVCMDataInputStream= new DataInputStream(aVCMInputStream);
 //crea il nome dell'output file, togliendo il suffisso .vcm e mettendo .txt
  int theInFileNameStringLength= theInFileNameString.length();
  String theOutFileNameString = theInFileNameString.substring(0,theInFileNameStringLength-4)+"1.txt";
 //crea il file per l'output e gli oggetti necessari alla scrittura
  File theTxtFile = new File(theOutFileNameString);
  theTxtFile.createNewFile();
  FileOutputStream theTxtFileOutStream= new FileOutputStream(theTxtFile);
  PrintStream outPrintStream = new PrintStream(theTxtFileOutStream);
 //legge le righe iniziali
 //minima distanza dal filo:
  float mindist=readFloat(aVCMDataInputStream,outPrintStream);
 //# punti di campionamento lungo x:
  short nx=readShort(aVCMDataInputStream,outPrintStream);
 //# punti di campionamento lungo y:
  short ny=readShort(aVCMDataInputStream,outPrintStream);
 //# punti di campionamento lungo z:
  short nz=readShort(aVCMDataInputStream,outPrintStream);
  float stepx=readFloat(aVCMDataInputStream,outPrintStream);
  float stepy=readFloat(aVCMDataInputStream,outPrintStream);
  float stepz=readFloat(aVCMDataInputStream,outPrintStream);
  float origx=readFloat(aVCMDataInputStream,outPrintStream);
  float origy=readFloat(aVCMDataInputStream,outPrintStream);
```

```
    float origz=readFloat(aVCMDataInputStream,outPrintStream);

    //matrice dei dati
    float[][][][] data= new float[4][nx][ny][nz];
    //ciclo di righe e colonne in lettura...
      for (int m=0; m<nz; m++)
       {for (int j=0; j<ny; j++)
        {for (int i=0; i<nx; i++)
         {for (int c=0; c<4; c++)
                {
                  data[c][i][j][m] = readFloat(aVCMDataInputStream,outPrintStream);
                }
                }
            }
        }
     //ciclo in scrittura
    for (int o=0; o<nz; o++)
       {for (int n=0; n<nx; n++)
           {for (int m=0; m<ny; m++)
                {
             outPrintStream.println(data[3][n][m][o]);
                }
        }}
  }
}
```

Palmanova, 12.06.2003

Consulente Fisico
dott. Francesco Montanari

Referente
dott.ing. Massimo Telesca

Responsabile del progetto
dott. Renato Villalta