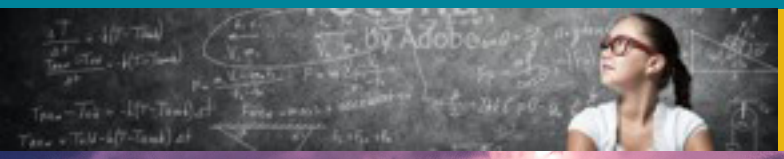
Allineamento LIDAR

Una procedura per controllare un telescopio



RESEARCH IN ACTION - RIA

RESEARCHINACTION.IT

Il LIDAR è un apparecchiatura per analizzare l'atmosfera. Il funzionamento è piuttosto semplice: un raggio laser viene inviato verticalmente nell'atmosfera, la luce rifratta e riflessa dall'atmosfera è captata da una serie di undici telescopi e analizzata per dedurre le condizioni e la composizione dell'atmosfera.

L'intento è quello realizzare una procedura che permetta di spostare un telescopio, passo passo, in modo automatico, fino a portarlo nella posizione migliore, quella in cui il segnale è più intenso!

10 Allineamento LIDAR - 05.19
Revisione 0 del 27.05.19

RiA - Research in Action

La parola ria in inglese significa estuario, in particolare (dalla definizione che ne da l'Oxford Living Dictonaries):

> A long, narrow inlet formed by the partial submergence of a river valley ... the rias or estuaries contain very peculiar ecosystems which often contain important amounts of fish ... (a causa della loro natura, le rias o estuari contengono ecosistemi molto particolari che spesso contengono grandi quantità di pesce - www.eurotomic.com/spain/the-rias-altas-in-spain.php)

quindi questo prodotto che sarà realizzato grazie all'attività di alternanza scuola-lavoro di alcuni studenti del liceo scientifico G.B.Grassi di Latina - **www.liceograssilatina.org** - sarà un luogo virtuale da esplorare dove *pescare* molto materiale per la didattica laboratoriale.

Fare scienza

La scienza non è solo identificabile con la formula, il modello, la teoria. In altre parole la scienza non rappresenta solo un corpo di conoscenze organizzate e formalizzate. La scienza è anche e fondamentalmente ricerca. Una ricerca volta a conoscere e a capire sempre più e sempre meglio come è fatto e come funziona questo nostro complicatissimo mondo.

Fare scienza si identifica con l'interrogarsi, con l'indagare ed esplorare fatti e cose. Questo tipo di lavoro i bambini lo fanno spontaneamente sin dalla loro nascita ma si perde nel corso del percorso scolastico. L'intervento educativo deve tener conto di ciò e fornire stimoli, occasioni e strumenti per far acquisire agli studenti capacità sempre più ampie e affinate per poter compiere questo lavoro di indagine mantenendo viva (o risvegliando) la curiosità cognitiva, la voglia di sapere e di scoprire, la fiducia di poter capire.

Pensare in senso creativo, in campo scientifico, significa aggredire i problemi, attivare processi vivi dei pensiero, alimentare l'evoluzione dinamica dell'intelligenza duttile, dell'esercizio dell'intuizione e dell'immaginazione, della capacità di progettare e formulare ipotesi, di controllare e verificare quanto prodotto e ricercato.

Per questo è necessario bandire forme di apprendimento consumate entro schemi rigidi di elaborazione del pensiero e puntare al recupero della congettura, dell'ipotesi, di una coscienza scientifica aperta a interrogare ogni problematica.

La società odierna deve far fronte ad un rinnovamento scientifico e tecnico accelerato in cui lo sviluppo delle conoscenze scientifiche e la creazione di prodotti di alta tecnologia *(hi-tech)*, come anche la loro diffusione subiscono un'accelerazione sempre più rapida.

È necessaria, quindi, una diffusione della conoscenza in genere ed è indispensabile promuovere una nuova cultura scientifica e tecnica basata sull'informazione e sulla conoscenza. E quanto più è solida la base di conoscenze scientifiche scolastiche, tanto più si può approfittare dell'informazione e della conoscenza scientifica e tecnica.

» https://www.facebook.com/Research-in-Action-341307966417448/
» https://www.youtube.com/channel/UC1PA7Zu78RUMBJnkaiOR8kA/

Sommario dei contenuti

Allineamento LIDAR - Una procedura per controllare un telescopio

Sommario dei contenuti

Blockly è un generatore di codice open source sostenuto da Google: https://developers.google.com/blockly/.

Materiale disponibile per questo laboratorio:

» il fascicolo (in formato PDF di circa 10MB): **http://researchinaction.it/wp-content/uploads/2019/05/10-Allineamento-lidar.pdf**;

» un ambiente in cui testare le procedure e le funzioni sviluppate nel corso di questo laboratorio e implementate con Blockly: **http://researchinaction.it/myblockly/lidar.html**;

» la libreria con le funzioni sviluppate in Python (ottenute direttamente da Blockly, in formato ZIP): **http://researchinaction.it/wp-content/uploads/2019/06/10-Allineamento-LIDAR.zip**.

Per il materiale didattico a supporto del fascicolo visitare anche la pagina Download del sito dedicato al progetto: **http://researchinaction.it/download/**.

Per i videotutorial è possibile visitare il canale YouTube del progetto: **https://www.youtube.com/channel/UC1PA7Zu78RUMBJnkaiOR8kA**.

Allineamento LIDAR

Una procedura per controllare un telescopio

1. Introduzione

Questo laboratorio è stato sviluppato da Matteo Cirenza, Veronica Civerchia, Lorenzo Fiorentini, Lisa Incollingo, Ilaria Lepori in collaborazione con Gianluigi Liberti (CNR-ISAC), il progetto è stato coordinato da Gualtiero Grassucci (Iss G.B. Grassi di Latina).

Il LIDAR *(Laser Imaging Detection and Ranging)* è un'apparecchiatura per analizzare l'atmosfera. Il funzionamento, almeno teoricamente, è piuttosto semplice: un raggio laser viene inviato verticalmente nell'atmosfera, la luce emessa è rifratta e riflessa dall'atmosfera a causa delle differenze di pressione, di umidità, di temperatura, della presenza di particelle in sospensione (l'aerosol). La radiazione riflessa (o meglio, *retro-riflessa*) è captata da una serie di undici telescopi e analizzata per dedurre le condizioni e la composizione dell'atmosfera. L'apparato è infatti dotato di fibre ottiche che trasferiscono il segnale luminoso alle sezioni di analisi spettrale e di fotorivelazione, che determinano quale tipo di particelle è presente nell'atmosfera.

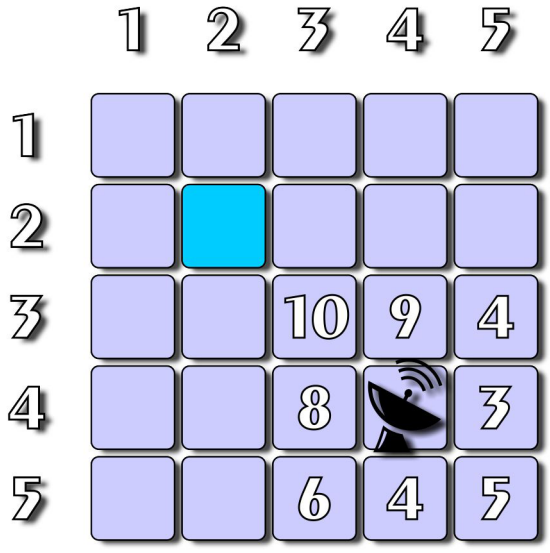
L'apparato a cui facciamo riferimento utilizza due raggi laser con una differente lunghezza d'onda:

» la radiazione verde (lunghezza d'onda pari a 532 nm) viene utilizzata per rilevare la temperatura e l'aerosol;
» il secondo, con lunghezza d'onda di 355nm, è un raggio ultravioletto (UV) che viene utilizzato per rilevare i segnali originati dal vapore acqueo e dall'azoto.

In assenza di disturbi ed errori il punto migliore in cui sistemare i telescopi è nelle immediate vicinanze dell'emettitore (il laser). Purtroppo la presenza di umidità, vento, polveri ecc. fanno sì che il segnale riflesso sia disturbato e deviato. Il nostro compito è quello di spostare il ricevitore (uno dei telescopi) per portarlo in una posizione in cui il segnale sia il migliore possibile.

Nel dettaglio, immaginiamo di avere una tabella (una matrice) di 21 x 21 caselle, ogni casella corrisponde a una posizione del telescopio, collocato inizialmente in una di queste posizioni. L'obiettivo è di portarlo nella casella in cui il segnale è migliore nel minor tempo possibile e quindi usando il minor numero possibile di passi. Un passo è lo spostamento del telescopio da una casella a una adiacente. Ovviamente vogliamo che la procedura sia automatizzata, senza intervento di un operatore.

Cerchiamo di chiarire bene qual è la situazione: nella figura qui accanto mostriamo una matrice ridotta, solo 5x5 caselle. Il telescopio attualmente si trova nella casella (4, 4): si intende la casella nella *quarta colonna* e nella *quarta riga*. I numeri che si trovano nelle caselle vicine indicano, in qualche unità di misura opportuna, il valore del segnale rilevato. La casella evidenziata in colore azzurro corrisponde alla posizione migliore per il rilevamento ma all'inizio della procedura non sappiamo dove essa sia, possiamo solo conoscere il valore del segnale nella casella attualmente occupata e in quelle vicine (così come mostrato nell'esempio). Quindi, l'intento è quello di spostare il telescopio, di casella in casella, fino a por-



tarlo nella posizione migliore!

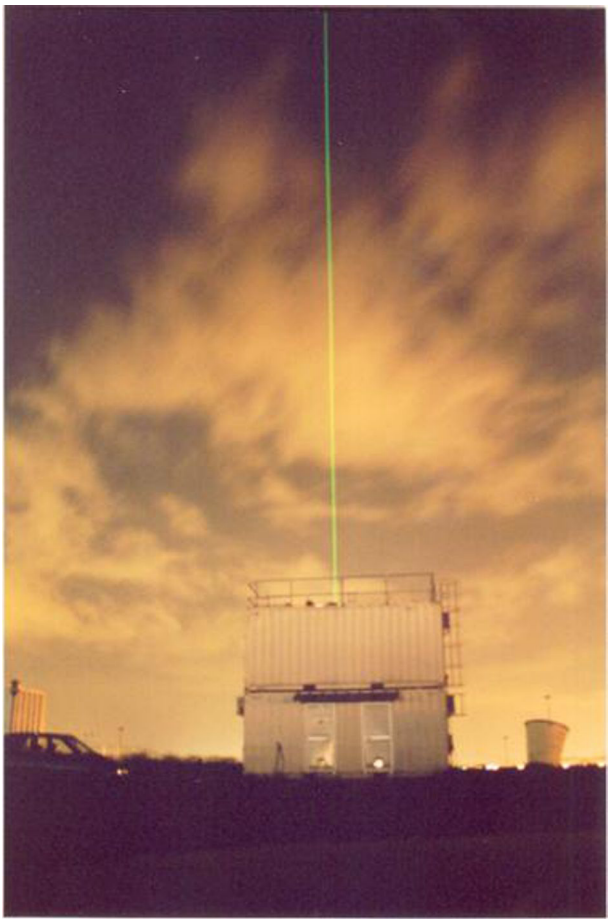
1.1. Prerequisiti

Per questo laboratorio è necessario:

» conoscere, almeno in maniera elementare, cos'è una matrice e saper identificare le caselle (le celle) di una matrice attraverso il numero di riga e di colonna.

1.2. Obiettivi

Il nostro obiettivo è quello di trovare un algoritmo in grado di allineare automaticamente il telescopio del LIDAR in modo tale che, ogni volta, esso occupi la posizione migliore per ricevere il segnale laser.

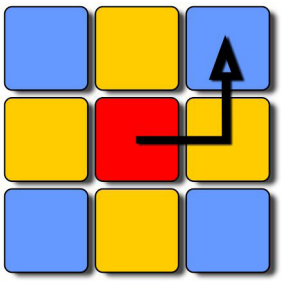


Il LIDAR RMR multicanale dell'Istituto per Studi sull'Atmosfera e sul Clima (CNR-ISAC) di Roma Tor Vergata.

2. Allineamento LIDAR

2.1. Algoritmo

Procediamo ora alla *costruzione* di un algoritmo che, preso un punto a caso in una matrice (in una tabella), ci porti nella casella, nella posizione, in cui il segnale è massimo.



Per comodità nel seguito, fissata una casella della matrice (in colore rosso nella figura qui accanto), indicheremo con il termine vicine le otto caselle che si trovano tutt'intorno a questa (in giallo e azzurro nella figura), con adiacenti le sole quattro caselle vicine che hanno un lato in comune con la casella considerata (in giallo nella figura) e con diagonali le quattro caselle, sempre vicine, che hanno solo un vertice in comune (in azzurro).

Gli spostamenti che può compiere il telescopio sono solamente quelli orizzontali e verticali: in altre parole, uno spostamento può avvenire solo in una delle caselle adiacenti (in alto, in basso, a destra, a sinistra), per spostarsi in una casella vicina diagonale sono necessari due spostamenti, due *passi*. Facendo riferimento alla figura qui sopra, se si vuole spostare il rilevatore dalla casella rossa alla casella diagonale situata in alto a destra si devono compiere due movimenti: per esempio, prima nella casella adiacente a destra e poi nella casella diagonale in alto a destra (la freccia, nella figura, indica il cammino).

Il Toolbox: http://researchinaction.it/wp-content/uploads/2018/11/00-Toolbox.pdf fornisce una definizione di algoritmo e i concetti fondamentali relativi alla realizzazione di una procedura in linguaggio naturale ma abbastanza rigoroso (cfr. 6. Algoritmi a pagina 23).

Un primo esempio

Cominciamo con un esempio. Prendiamo una matrice più piccola di 5 righe e 5 colonne. Ricorda che il tuo scopo è quello di arrivare al massimo assoluto, ovvero nella casella che ha il valore più grande, con meno spostamenti possibili e tenendo conto che puoi conoscere solamente i valori delle 8 caselle intorno a quelle di partenza. Nella figura in basso in questa pagina c'è la matrice di esempio, il valore di ogni casella, in opportune unità di misura indica la qualità, l'intensità, del segnale (maggiore è il numero, migliore è il segnale ricevuto). Ricorda che puoi spostarti solo nelle caselle direttamente adiacenti a quella in cui ti trovi e quindi se devi di spostarti in diagonale devi compiere due movimenti (uno in orizzontale e uno in verticale).

Per iniziare, posizionati nella casella di coordinate *(3, 4)*, cioè nella casella che si trova nella terza colonna e nella quarta riga (è colorata in giallo).

Adesso spostati di una casella in modo da avvicinarti a quella corrispondente al massimo.

	1	2	3	4	5
1	10	11	11	5	3
2	9	15	12	7	4
3	10	13	10	9	4
4	8	7		6	3
5	4	5	6	4	5

Tieni presente il modo con cui hai scelto la casella dove spostarti e ricorda che devi spostarti solo conoscendo il valore del segnale nelle otto caselle vicine a quella attuale!

Riprova, spostati ancora, di casella in casella, fino ad arrivare a quella con il massimo. Come hai scelto la direzione del movimento ogni volta? Puoi formalizzare in una regola il processo di scelta?

In pratica, stiamo cercando un algoritmo che

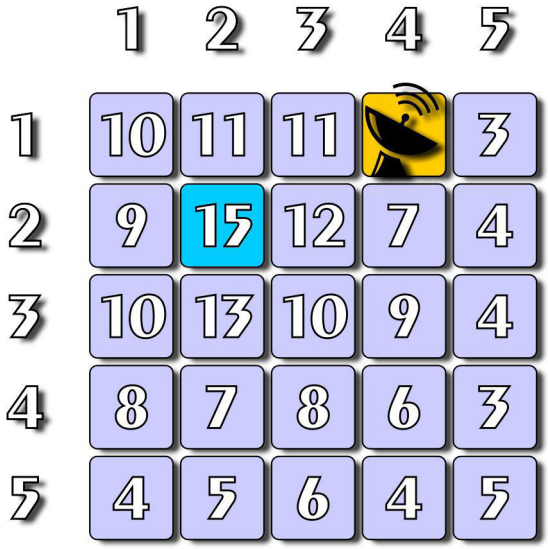
consenta di arrivare al massimo e che possa essere eseguito in modo automatico.

C'è un'altra cosa di cui tener conto: come hai deciso di fermarti?

La procedura deve terminare dopo un numero finito di passi ma tieni conto che puoi conoscere solamente il contenuto della casella in cui ti trovi, delle caselle adiacenti e di quelle in diagonale: non puoi avere una visione dell'intera tabella e devi quindi decidere di fermarti solamente con le informazioni di cui disponi (la cella in cui si trova attualmente il telescopio e le 8 caselle intorno)!

Riproviamoci

Posizionati adesso nella casella *(4, 1)*, quarta colonna e prima riga, e ricomincia, tenendo conto della regola che hai ipotizzato in precedenza.



Non ti focalizzare, però, sulla ricerca del massimo ma sul *perchè* dei tuoi movimenti, sul *cosa*, dunque, ti spinge nello spostarti nella casella successiva.

Non disperare la strada è sicuramente in salita, ma una volta arrivati in cima la soddisfazione sarà molta!

Il Toolbox: http://researchinaction.it/wp-content/uploads/2018/11/00-Toolbox.pdf fornisce una definizione di algoritmo e i concetti fondamentali relativi alla realizzazione di una procedura in linguaggio naturale ma abbastanza rigoroso (cfr. 6. Algoritmi a pagina 23).

Reputi ci siano delle modifiche da fare alla regola che ti sei dato? Sei arrivato ad avere una procedura che schematizzi e che guidi il tuo percorso alla ricerca del massimo, tenendo conto delle possibili problematiche che potresti incontrare?

Se ad esempio, per assurdo, il tuo criterio è quello di spostarti verso il numero pari più grande, cosa succede se ti circondano solo numeri dispari?

2.2. La procedura

È giunto il momento di scrivere la procedura. Per aiutarti cercheremo di dividerla in tre *step* parziali, che poi dovremo *assemblare* per avere la soluzione del problema.

La casella vicina con il valore maggiore

Il primo passo è quello di scrivere alcune istruzioni che permettano di determinare la casella *vicina* che ha il valore maggiore.

Supponi di essere nella casella *(x, y)* dove, come al solito, *x* è la riga e *y* è la colonna. Formalizza le istruzioni per determinare quale tra le caselle vicine (sia in diagonale che adiacenti) ha il valore maggiore.

Tieni conto che, se abbiamo chiamato *M* la matrice, il valore della casella in cui ti trovi è *M(x, y)*. Ricorda anche che, come negli esempi che abbiamo mostrato in precedenza, la casella adiacente posta immediatamente sopra ha coordinate (x, y-1) mentre quella in diagonale in basso a destra, per esempio, ha coordinate *(x+1, y+1)*.

Specifica quali sono le informazioni che questa piccola, parziale, procedura, deve resti-

tuire, informazioni che ci saranno utili per progettare l'algoritmo completo. Concentrati anche sui casi particolari, per esempio se la casella attuale è sul margine della matrice o in un angolo. Controlla che le tue istruzioni siano corrette anche in questi casi.

A questo punto dovresti avere una funzione che, una volta indicato l'input corretto, restituisce la giusta informazione relativamente alla casella vicina con valore maggiore. qualcosa del tipo:

```
... output ... = VicinaMaggiore( ... input ...)
```

Se non sei sicuro di quanto fatto, prima di proseguire puoi confrontare il lavoro fatto fino a qui con la soluzione che proponiamo nel paragrafo **La casella vicina con il valore maggiore a pagina 15**.

Un piccolo passo per il nostro telescopio

Ora cerchiamo di far muovere il telescopio.

Scrivi alcune istruzioni, in modo quanto più formale e rigoroso possibile, per spostare il telescopio da una certa casella di coordinate (x, y) a una casella vicina, diciamo di coordinate (xm, ym).

Tieni conto della differenza tra caselle adiacenti e diagonali e dei limiti che abbiamo sugli spostamenti diagonali **(cfr. 2.1 Algoritmo a pagina 7)**.

Precisa cosa deve restituire la funzione che sposta il telescopio (l'output).

Ora dovremmo avere una seconda funzione che, con il giusto input, sposta il telescopio in una delle caselle adiacenti. Una funzione del tipo:

```
... output ... = Passo( ... input ...)
```

che fa compiere il *piccolo passo* di cui parlavamo nel titolo di questo paragrafo.

Anche in questo caso, se lo desideri, puoi confrontare la tua soluzione con quella che proponiamo nel paragrafo **Un piccolo passo per il nostro telescopio a pagina 17**.

Assemblaggio

Non ci resta che *mettere insieme* i *mattoncini* che abbiamo preparato per costruire la procedura completa. Cerca adesso di mettere insieme tutte le considerazioni che hai fatto fino ad ora per scrivere la procedura completa, quella che, a partire da una casella qualsiasi, sposta il telescopio nella posizione in cui il segnale è migliore.

Innanzitutto precisa le informazioni che saranno necessarie a questa procedura: l'input.

Questo passo dovrebbe essere il più facile.

Adesso concentriamoci sul cuore della procedura, ricorda che lo scopo è quello di arrivare ad un algoritmo, seppure semplice, ovvero ad una procedura che, se applicata, ci porti automaticamente, con il minor numero di passaggi, al massimo.

Formalizza le istruzioni che devono essere ripetute più volte, a ogni passo, e la condizione da soddisfare per concludere l'algoritmo quando il massimo è raggiunto.

Dobbiamo cercare di realizzare una serie di istruzioni che possono essere eseguite una dopo l'altra anche da chi non è a conoscenza dei dettagli del problema e ha a disposizione solo la matrice

di esempio, carta e penna. Puoi usare le due piccole funzioni che hai scritto in precedenza per determinare i risultati parziali o le singole operazioni di movimento. Al termine dovremmo avere una funzione del tipo:

```
... output ... = Allineamento ( ... input ...)
```

che, con le giuste informazioni, automaticamente, sposta il telescopio nella posizione ideale.

Qualche consiglio

Ogni istruzione deve essere quanto più semplice (non ambigua) possibile. Se un'istruzione è troppo complessa o richiede più (troppe) operazioni per essere eseguita prova a immaginare di *scomporla* in istruzioni più semplici. Per esempio, se hai bisogno di determinare il valore più grande presente nelle caselle *vicine*, puoi pensare di scrivere a parte una piccola serie di istruzioni che facciano questo per te e richiamare questa *funzione* nella procedura principale.

Le istruzioni devono essere generali, non fare affidamento a quello che vedi, ma a quello che l'apparato può misurare in modo indipendente.

Non dare nulla per scontato, controlla che, in una situazione reale, ogni informazione usata dalla procedura che stai scrivendo sia effettivamente disponibile.

Analizza eventuali casi particolari e assicurati che le istruzioni che hai ipotizzato funzionino in modo corretto e coerente anche nei casi particolari. Per esempio, se a un certo punto del suo cammino il telescopio si trova in una casella al margine della matrice, o in un angolo, va tutto bene?

2.3. La matrice completa

Adesso, tornando al nostro problema, una volta capito come si deve procedere, non dovrebbe essere difficile ricominciare con la matrice 21x21. Infatti, se il metodo che hai progettato per la matrice più piccola è generale dovrebbe funzionare anche nel caso di quella più grande.

Utilizza la procedura che hai formalizzato prima e qualora fosse necessario modificala: scegli una casella a caso della matrice e cerca di seguire le istruzioni che hai scritto per arrivare al massimo.

Ricordati che il tuo algoritmo deve far fronte a qualsiasi problema si potrebbe presentare, quindi non essere superficiale!

Prova l'algoritmo più volte, partendo da diversi punti della matrice, tieni conto del numero dei passaggi che compi sia sull'asse delle x che quello delle y e cerca sempre di ottimizzare il numero di passaggi da fare.

Se l'algoritmo che hai progettato funziona correttamente sulla matrice di esempio che trovi nella pagina successiva, siamo davvero a buon punto: hai già fatto un ottimo lavoro!

Se sei soddisfatto dei risultati raggiunti e vuoi controllare la bontà della tua soluzione puoi confrontare il tuo algoritmo con la procedura che proponiamo nelle soluzioni (cfr. 3.2 La procedura a pagina 15.

2.4. Il problema vero e proprio

Ora cerchiamo di mettere alla prova il tuo algoritmo su una matrice costruita in modo più realistico, più vicina alle misure realistiche del segnale rilevato dal telescopio.

In matematica, una funzione gaussiana è una funzione della seguente forma:

$$g(x) = \frac{1}{\sigma \cdot \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

dove μ è il valore atteso e σ^2 la varianza o, con parole diverse, è lo scarto quadratico medio.

Mostra il caratteristico andamento di una distribuzione normale di probabilità, dove μ è la media delle misure (e anche la misura più probabile) mentre lo scarto quadratico medio, in un certo senso, determina la qualità delle misure effettuate.

Supponiamo di dover fare delle misurazioni della stessa grandezza con uno strumento, otterremmo sicuramente risultati differenti a causa dell'imprecisione degli strumenti e dell'incidenza sulla misura delle condizioni ambientali. Se costruissimo un grafico con il valore delle misure sull'asse delle ascisse e la frequenza con cui ogni misura si è presentata otterremo, in alcune situazioni, quella che abbiamo chiamato distribuzione normale: una curva con un massimo corrispondente alla media i cui valori tendono via via ad annullarsi allontanandosi dalla media. Quindi il punto corrispondente alla media è anche il valore della misura più probabile, la misura più vicina a quella reale.

0	1	2	4	6	5	4	3	2	1	0	1	0	2	4	3	2	1	2	3	4
2	6	5	7	8	7	6	5	4	3	2	1	3	5	6	5	4	3	7	5	8
3	7	8	9	10	11	8	10	9	6	4	5	6	8	9	10	7	6	8	9	10
2	6	11	12	13	12	13	14	16	15	14	12	10	11	13	11	12	10	11	12	13
4	9	13	17	14	15	20	18	17	19	18	16	14	15	20	19	16	17	18	19	20
8	11	12	19	20	21	24	25	20	29	28	27	24	22	21	25	20	27	28	26	22
7	10	14	18	22	26	30	32	34	38	32	30	31	28	29	30	32	35	34	38	36
6	9	13	20	28	30	39	40	43	47	44	37	45	34	38	37	39	44	45	42	40
4	8	12	16	26	31	35	46	50	51	52	54	52	48	47	58	59	57	52	49	39
5	6	11	20	24	33	42	47	54	58	59	57	55	52	59	60	62	54	59	48	44
1	7	10	24	30	35	39	48	51	69	68	66	60	61	64	65	66	68	57	50	45
3	5	13	23	28	29	41	49	54	67	72	74	76	73	78	79	77	69	53	47	40
2	9	16	17	24	30	44	50	55	62	72	80	81	80	88	89	70	65	52	48	42
1	10	11	19	26	34	46	51	54	65	75	81	95	98	90	86	75	70	57	45	39
2	8	14	20	27	33	43	53	58	60	78	86	94	99	97	85	76	69	51	44	38
3	9	12	17	29	32	40	51	59	64	74	84	96	94	92	82	73	70	50	42	36
4	8	13	15	20	30	39	49	54	69	75	88	87	80	87	83	70	68	55	46	32
5	7	10	14	27	31	37	48	50	68	79	75	74	76	70	77	78	67	56	41	38
3	6	9	13	24	28	32	41	57	65	64	67	62	61	62	65	60	66	60	49	41
4	8	11	17	22	27	31	49	53	54	56	51	50	54	58	57	59	64	61	46	39
1	5	10	19	20	25	30	47	46	50	48	49	41	52	50	47	46	39	45	40	36

Qui a lato una matrice 21x21 caselle che simula un segnale disturbato dalle condizioni atmosferiche. La casella con il valore più alto è evidenziata in azzurro. Gli altri colori servono solo per aiutare a orientarsi sulla tabella.

La gaussiana è quindi lo strumento matematico perfetto per simulare un segnale disturbato dalle condizioni ambientali. La useremo per costruire una tabella (una matrice) che simuli la dispersione del segnale retro-riflesso e testare la procedura che realizzeremo.

Costruiamo la matrice

Nel nostro caso il segnale è disperso, disturbato, sporcato, in due direzioni (le righe e le colonne della tabella, le due direzioni x e y della superficie terrestre nei pressi del rilevatore) per cui abbiamo bisogno di una funzione gaussiana che tenga conto dello spostamento sia in direzione dell'asse delle ascisse che in quello delle ordinate:

$$g(x,y)=\frac{1}{\sqrt{\sigma_x^2+\sigma_y^2}\cdot\sqrt{2\pi}}e^{-\frac{[(x-\mu_x)^2+(x-\mu y)^2]}{2\cdot(\sigma_x^2+\sigma_y^2)}}$$

dove (μ_x, μ_y) sono le coordinate del valore atteso (della media, della misura più probabile) mentre

$$(x-\mu_x)^2+(x-\mu y)^2$$

è il quadrato della distanza di un generico punto (x, y) dal punto (μ_x, μ_y) e, infine:

$$\sigma_x^2+\sigma_y^2$$

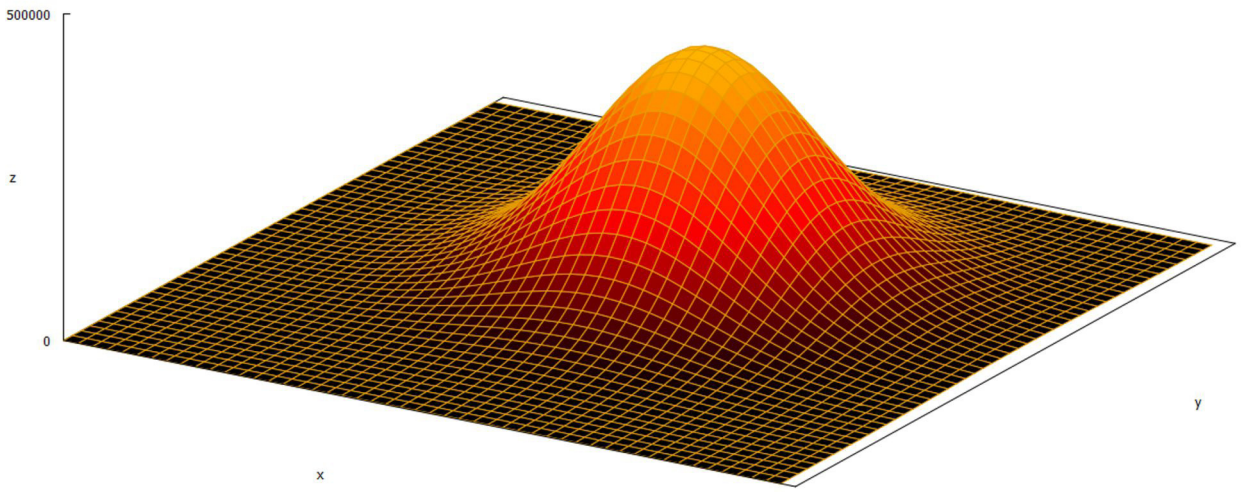
è la varianza che tiene conto, in questo caso, dell'errore sia sull'ascissa che sull'ordinata.

In realtà, il segnale reale, come detto in precedenza, è disturbato quindi dobbiamo sporcare la funzione con un errore in modo da avvicinare quanto più possibile la nostra simulazione alla situazione reale.

Sostanzialmente si tratta di aggiungere ai valori della gaussiana per così dire *standard* un valore casuale determinato da una formula del tipo

$$random\cdot NOISEMAX\cdot\sqrt{g(x,y)}$$

dove *random* è un numero casuale compreso tra 0 e 1 e *NOISEMAX* è un numero che può valere 0.5, 1.0 o 1.5 e sta a indicare quanto il segnale riflesso è disturbato. La funzione modificata, sporcata, è dunque questa:



Il grafico della funzione è stato realizzato con xMaxima, applicazione Open Source che si può scaricare qui: maxima.sourceforge.net/. In paricolare, dal sito researchinaction.it è possibile scaricare l'esercizio svolto usando xMaxima: ...

$$g_{reale}(x,y) = g(x,y) + random \cdot NOISEMAX \cdot \sqrt{g(x,y)}$$

L'aspetto di questa funzione è mostrato in fondo alla pagina precedente.

Costruisci una matrice 21 x 21 elementi, in ogni casella della matrice inserisci il valore della funzione modificata, considerando come x la colonna della casella e come y la riga.

Probabilmente ti sarà utile un foglio elettronico o un'applicazione CAS *(Computer Algebra System)* per velocizzare i calcoli. Inoltre, visto il valore piuttosto piccolo della gaussiana, soprattutto lontano dal massimo, converrà moltiplicare i valori che hai ottenuto per 10^4 o 10^3.

Scegli un punto in cui il segnale è migliore, la casella in cui si trova il valore maggiore: quello è il nostro obiettivo, la casella in cui portare il telescopio! Puoi simulare differenti situazioni con valori più alti per la varianza σ_x e σ_y che indicheranno una qualità delle misure peggiore e una maggiore dispersione del segnale.

Per ogni eventualità, puoi trovare una matrice costruita così come abbiamo indicato in questo stesso fascicolo **a pagina 11**.

Applica più volte la procedura che hai scritto alla matrice, realistica, che hai costruito partendo ogni volta da una casella scelta a caso. Controlla che, dopo un certo numero di passi, il telescopio sia nella casella dove si trova il segnale migliore e che la procedura si interrompa proprio quando è in quella posizione.

Se ogni volta, al termine della procedura, il telescopio è al *posto giusto*, abbiamo raggiunto il nostro obiettivo. *Good job!*

Precauzioni

Probabilmente, per evitare problemi, soprattutto la comparsa di *falsi massimi*, è meglio costruire la matrice come media di tre matrici differenti.

3. Algoritmo

3.1. Un primo esempio

Riprendiamo il primo esempio che abbiamo proposto **(cfr. Un primo esempio a pagina 7)**, partiamo dalla casella *(3, 4)*.

Adesso spostati di una casella in modo da avvicinarti a quella corrispondente al massimo.

Sempre supponendo di non poter conoscere altro che i valori del segnale nelle caselle *vicine*, l'unica possibilità è quella di determinare, tra queste posizioni vicine, quella che ha il valore maggiore: in questo caso la casella *(2, 2)* e poi spostarsi.

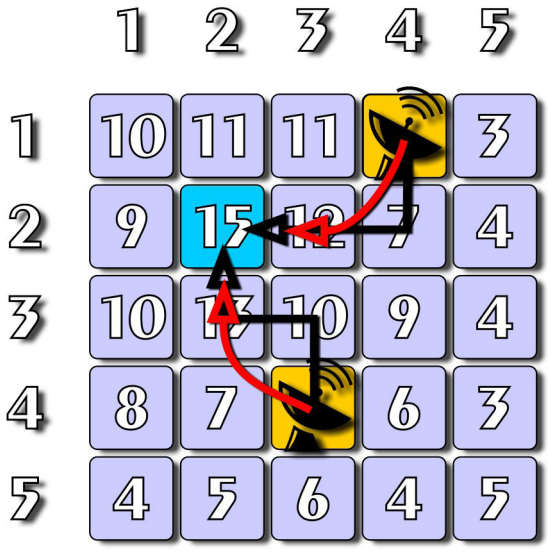
Riprova, spostati ancora, di casella in casella, fino ad arrivare a quella con il massimo. Come hai scelto la direzione del movimento ogni volta? Puoi formalizzare in una regola il processo di scelta?

Applichiamo la stessa regola: cerchiamo la casella vicina con il valore maggiore e poi ci spostiamo proprio in quella. La casella vicina con il valore più alto è la (2, 2), ci spostiamo. È bene notare che non sappiamo ancora che questa casella è quella giusta, la posizione migliore.

C'è un'altra cosa di cui tener conto: come hai deciso di fermarti?

Proviamo ancora una volta ad applicare la stessa regola. La casella con il valore più alto è la (2, 3) ma ... il valore è 13, minore di quello della casella in cui siamo posizionati. Quindi la condizione per fermarsi potrebbe essere: se il valore della casella attuale è maggiore di quelle vicine, fermati: sei arrivato nella posizione migliore!

Nella figura qui sotto il percorso è indicato con una freccia di colore nero. La freccia in rosso, invece, mostra il percorso che avremmo dovuto seguire senza la limitazione degli spostamenti solo orizzontali o verticali.



Riproviamoci

Posizionati adesso nella casella *(4, 1)*, quarta colonna e prima riga, e ricomincia, tenendo conto della regola che hai ipotizzato in precedenza.

Visto che sembrava funzionare, proviamo a seguire lo stesso procedimento. La casella vicina con il valore maggiore è la *(3, 2)*, il valore di questa casella *(12)* è maggiore della casella attuale *(5)*, quindi ci spostiamo.

Ancora. La casella vicina alla *(3, 2)* con il valore maggiore è la *(2, 2)*, valore che è più grande di quello attuale, quindi ci spostiamo ancora.

Infine: la casella vicina alla *(2, 2)* con il valore più grande è la *(2, 3)* ma non supera il valore della casella attuale: siamo arrivati!

Reputi ci siano delle modifiche da fare alla regola che ti sei dato? Sei arrivato ad avere una procedura che schematizzi e che guidi il tuo percorso alla ricerca del massimo, tenendo conto delle possibili problematiche che potresti incontrare?

L'algoritmo quindi deve svilupparsi in più fasi:

» La prima è quella della *ricerca* (la *vicina maggiore*): una volta che mi sono posizionato in una casella della matrice determino quale casella intorno ha il valore maggiore.
» La seconda è quella del movimento (*passo*): potendomi spostare unicamente in orizzontale e in verticale, il numero di spostamenti che faccio in un passaggio varia se la casella in cui mi devo spostare è adiacente o diagonale; in questo ultimo caso, infatti, bisogna compiere due movimenti: uno nella casella adiacente e uno, immediatamente successivo, nella diagonale rispetto a quella di partenza.
» L'ultima fase infine deve essere quella che mi indichi come, dove e perché fermarmi (*allineamento*): quest ultimo passaggio in realtà dovrebbe essere il più semplice, infatti se la casella in cui sono risulta avere il valore maggiore al confronto con tutte quante le 8 intorno, allora la casella in cui mi trovo è la casella del massimo.

Nella figura nella pagina precedente i due percorsi anche per questo secondo esempio.

3.2. La procedura

Probabilmente ora è arrivato il momento di essere un po' più formali.

La casella vicina con il valore maggiore

Supponi di essere nella casella *(x, y)* dove, come al solito, *x* è la riga e *y* è la colonna. Formalizza le istruzioni per determinare quale tra le caselle vicine (sia in diagonale che adiacenti) ha il valore maggiore.

Come già detto, se la posizione iniziale è *(x , y)*, le otto caselle che ho intorno saranno: *(x−1, y−1)* in alto a sinistra, *(x, y−1)* adiacente in alto, e così via ...

Specifica quali sono le informazioni che questa piccola, parziale, procedura, deve restituire, informazioni che ci saranno utili per progettare l'algoritmo completo. Concentrati anche sui casi particolari, per esempio se la casella attuale è sul margine della matrice o in un angolo. Controlla che le tue istruzioni siano corrette anche in questi casi.

Di seguito mostriamo la funzione così come l'abbiamo formalizzata seguendo le idee che abbiamo sperimentato nel corso del laboratorio (abbiamo indicato con righe il numero di righe della matrice e con colonne il numero di colonne)

```
VicinaMaggiore(x, y)
      Determina la casella, tra le otto vicine della casella specificata,
      quella che ha il valore più grande
      max = - inf
      x_m = x-1
      y_m = y-1
      se x-1 > 0 e y-1 > 0 allora
            casella diagonale in alto a sinistra
            se M(x-1, y-1) > max allora
                  max = M(x-1, y-1)
                  x_m = x-1
                  y_m = y-1
      se y-1 > 0 allora
            casella adiacente sopra
            se M(x, y-1) > max allora
                  max = M_2(x, y-1)
                  x_m = x
                  y_m = y-1
      se x+1 < colonne e y-1 > 0 allora
            casella diagonale in alto a destra
            se M(x+1, y-1) > max allora
                  ...
      se x+1 < colonne allora
            casella adiacente a destra
            se M(x+1, y) > max allora
                  ...
      se x+1 < colonne e y+1 < righe allora
            casella diagonale in basso a destra
            se M(x+1, y+1) > max allora
                  ...
      se y+1 < colonne allora
            casella adiacente in basso
            se M(x, y+1) > max allora
                  ...
      se x-1 > 0 e y+1 < righe allora
            casella diagonale in basso a sinistra
            se M(x-1, y+1) > max allora
                  ...
      se x-1>0 allora
            casella adiacente a destra
            se M(x-1, y) > max allora
                  ...
ritorna (x_m, y_m)
```

Il valore massimo viene individuato usando un piccolo trucco: si pone la variabile max pari a un valore enorme ma negativo con l'assegnazione *max = - inf* così che il primo confronto è sicuramente vero e quindi il massimo diventa pari al valore della prima casella raggiungibile.

Negli ultimi due controlli (sulla casella diagonale in basso a sinistra e su quella adiacente a destra) sono stati omessi (e sostituite dai puntini di sospensione) le istruzioni di aggiornamento del valore del massimo e delle coordinate corrispondenti.

I controlli del tipo *se x-1 > 0 e y-1 > 0 allora* ... fanno sì che non si acceda a caselle della matrice inesistenti. Infatti, se *x = 1* oppure *y = 1* (cioè ci si trova in una casella sul margine sinistro o su quello superiore) allora *x-1* o *y-1* darebbero come valore zero: fuori dalla matrice!

La funzione restituisce le coordinate della casella con il valore massimo perchè possano essere usate nel seguito.

Per comodità di lettura, abbiamo evidenziato in colore azzurro quelle che sarebbero le *parole chiave* del nostro linguaggio formale e in verde i commenti (che non sono istruzioni, ma solo spiegazioni delle istruzioni vere e proprie).

Un piccolo passo per il nostro telescopio

Scrivi alcune istruzioni, in modo quanto più formale e rigoroso possibile, per spostare il telescopio da una certa casella di coordinate (x, y) a una casella vicina, diciamo di coordinate (xm, ym).

Supponiamo di conoscere la casella in cui spostare il telescopio: se è adiacente il movimento è solo orizzontale o verticale, se è diagonale eseguo due spostamenti, prima in verticale e poi in orizzontale.

```
Passo(x, y, x_m, y_m)
      Si sposta dalla casella (x, y) alla casella (x_m, y_m)
      se x = x_m oppure y = y_m allora
            si sposta in una casella adiacente
            x = x_m
            y = y_m
      altrimenti
            si sposta due volte per andare in una casella in diagonale
            y = y + (y_m - y) spostamento verticale
            x = x + (x_m - x) spostamento orizzontale
ritorna (x, y)
```

Il controllo se *x = x_m oppure y = y_m allora ...* è vero se le due caselle sono adiacenti, falso in caso contrario. Le istruzioni del tipo *y = y + (y_m - y)* usano un altro piccolo trucco per assegnare alla variabile *y* la riga della casella destinazione in un colpo solo, senza scrivere istruzioni diverse nel caso la casella destinazione sia *sopra* o *sotto*. Per esempio, se la casella attuale è $(2, 4)$ e quella destinazione è $(2, 3)$, quindi sopra, l'istruzione diventa $y = 4 + (3 - 4) = 4 + (-1) = 3$: corretto!

Precisa cosa deve restituire la funzione che sposta il telescopio (l'output).

La funzione restituisce le coordinate della nuova casella attuale, in modo da consentire alla procedura di proseguire il percorso.

Sul blog Research in Action è possibile utilizzare le funzioni che sono state sviluppate nel corso del laboratorio e implementate, in seguito, con Blockly: http://researchinaction.it/myblockly/lidar.html
Blockly è un generatore di codice open source sostenuto da Google: https://developers.google.com/blockly/.

Innanzitutto precisa le informazioni che saranno necessarie a questa procedura: l'input.

Alla procedura è necessario comunicare la matrice *M* e la casella di partenza *(x, y)*. Non c'è bisogno di altro.

Formalizza le istruzioni che devono essere ripetute più volte, a ogni passo, e la condizione da soddisfare per concludere l'algoritmo quando il massimo è raggiunto.

```
Allineamento(M, x, y)
      Sposta il telescopio nella casella corrispondente al massimo
      M è la matrice e x,y le coordinate della casella di partenza
      ripeti
            (x_m, y_m) = VicinaMaggiore(x, y)
            se M(x_m, y_m) <= M(x, y) allora
                  abbiamo trovato il massimo, fine
                  ritorna (x, y)
            altrimenti
                  (x, y) = Passo(x, y, x_m, y_m)
```

La procedura sostanzialmente ripete sempre le stesse poche istruzioni: determina la casella vicina con il valore più grande (e per farlo usa la funzione *VicinaMaggiore(...)* che abbiamo progettato in precedenza) e controlla se il valore di questa casella è minore di quello attuale, in caso di risposta affermativa siamo già nella casella obiettivo e il ciclo è interrotto restituendo le coordinate proprio della casella giusta (che, tra l'altro, è quella in cui si trova il telescopio attualmente).

Se, invece, c'è ancora un valore più grande di quello attuale, il telescopio è spostato nella casella corrispondente (e, anche in questo caso, si usa la funzione *Passo(...)* costruita nel corso del laboratorio).

In pratica: se la casella attuale risulta avere un valore maggiore al confronto con tutte quante le otto intorno, allora la casella in cui mi trovo è la casella del massimo!

4. Esercizi

4.1. Il percorso

Così come è stata strutturata la funzione *Allineamento(...)* restituisce solo la posizione finale. Sarebbe utile, però, per ulteriori controlli e miglioramenti, conoscere l'intero percorso fatto.

Modifica la funzione *Allineamento(...)* in modo che costruisca una lista contenente le coordinate delle caselle attraversate, aggiungendo ogni volta le coordinate della casella in cui viene spostato il telescopio. La prima casella dovrebbe essere quella di partenza e l'ultima quella di arrivo.

A questo punto la funzione potrebbe restituire questa lista e non solo le coordinate del punto di arrivo.

4.2. Ottimizzazione

La procedura che abbiamo formalizzato funziona correttamente ma è molto dispendiosa in termini di tempo se si considera che la misura del valore del segnale in una certa posizione richiede più o meno un minuto. Infatti, nella realtà, ogni volta che accediamo alla matrice con un'istruzione del tipo $M(x, y)$ richiediamo la misura dello stato del segnale in una certa posizione. Si capisce che, con queste informazioni, la funzione *VicinaMaggiore(...)* richiede, nel caso peggiore, 16 minuti circa (ci sono sedici accessi alla matrice).

Supponendo però che l'atmosfera, nel periodo di utilizzo dell'apparato, sia abbastanza stazionaria, potremmo cercare di riutilizzare i valori appena misurati nelle caselle vicine senza richiedere ogni volta una nuova rilevazione allo strumento.

Cerca di modificare la procedura aggiungendo una seconda matrice, diciamo M_2, che inizialmente sarà *vuota*, in cui memorizzare (mettere da parte) i valori di $M(x, y)$ via via che vengono misurati. A questo punto si può correggere la procedura *VicinaMaggiore(...)* in modo che ricavi i valori delle caselle vicine dalla matrice M_2 quando possibile e richieda una nuova rilevazione solo quando il valore della casella corrispondente non è ancora stato misurato.

Blockly Google
A Visual Programming Language

Sul blog Research in Action è possibile utilizzare le funzioni che sono state sviluppate nel corso del laboratorio e implementate, in seguito, con Blockly: http://researchinaction.it/myblockly/lidar.html
In particolare, ci sono alcune funzioni dedicate alla gestione delle matrici che possono essere usate come supporto per le procedure progettate in questo laboratorio.
Blockly è un generatore di codice *open source* sostenuto da Google: https://developers.google.com/blockly/.

4.3. Coding

Provare a codificare le procedure con il generatore di codice Blockly. Non c'è la possibilità nativa in Blockly di gestire le matrici per cui si dovranno progettare funzioni apposite o usare la libreria progettata appositamente per questo laboratorio.

5. La libreria

Sul blog Research in Action è possibile utilizzare le funzioni che sono state sviluppate nel corso del laboratorio e implementate, in seguito, con Blockly: http://researchinaction.it/myblockly/lidar.html
In particolare, ci sono alcune funzioni dedicate alla gestione delle matrici che possono essere usate come supporto per le procedure progettate in questo laboratorio.
Blockly è un generatore di codice open source sostenuto da Google: https://developers.google.com/blockly/.

Contestualmente alla soluzione del problema, nel corso del laboratorio le funzioni realizzate sono state implementate mediante Blockly. Attualmente il generatore di codice di Google non supporta le matrici (anche se è previsto uno sviluppo in tal senso: **https://developers.google.com/blockly/guides/modify/future**) per cui, per *maneggiare* la matrice del problema abbiamo dovuto progettare e realizzare alcune funzioni apposite. In questo *capitoletto* presentiamo la libreria e la sua struttura.

5.1. Variabili globali e impostazioni

La libreria fa uso di alcune variabili globali che impostano le dimensioni della matrice e i parametri della funzione gaussiana usata per simulare il segnale (e il disturbo dovuto alle condizioni dell'atmosfera):

» *righe, colonne*: il numero di righe e di colonne della matrice;
» *mux, muy*: il valore atteso lungo l'asse delle ascisse e lungo l'asse delle ordinate e quindi, in pratica, la colonna e la riga della casella della matrice in cui, teoricamente, si dovrebbe trovare il segnale migliore;
» *sigmax, sigmay*: la radice della varianza lungo l'asse delle ascisse (colonne) e lungo l'asse delle ordinate (righe);
» *noisemax*: il vallore massimo da attribuire al disturbo;
» *scala*: un fattore per cui moltiplicare i valori della distrubuzione normale del segnale.

Tutte le variabili globali sono impostate nella stessa funzione:

```
InizializzazioneParametri(righe, colonne, mux, muy, sigmax, sigmay,
      noisemax, scala)
```

5.2. Funzioni per la gestione delle matrici

Le matrici sono implementate come liste di righe, ogni riga a sua volta è una lista di elementi. Le posizioni nella matrice sono *mappate* su un sistema di riferimento che ha origine in alto a sinistra, l'asse delle ascisse che va da sinistra verso destra e quello delle ordinate dall'alto verso il basso.

Convertire una matrice in una stringa di testo

Converte una matrice in una stringa di testo in cui gli elementi sono separati da virgole e le righe racchiuse tra parentesi quadre.

```
MatriceTesto(m)
```

dove *m* è la matrice. Restituisce una stringa di testo costruita come descritto sopra.

Accedere agli elementi della matrice

Recupera il valore di un elemento della matrice.

```
ElementoMatrice(m, x, y)
```

dove *m* è la matrice e *x, y* rispettivamente la colonna e la riga dell'elemento del quale recuperare il valore. Restituisce il valore dell'elemento specificato.

Imposta il valore di un elemento della matrice.

```
ImpostaElementoMatrice(m, x, y, valore)
```

dove *m* è la matrice, *x, y* rispettivamente la colonna e la riga dell'elemento di cui impostare il valore e *valore* è proprio il valore da assegnare a quell'elemento.

Numero di righe e colonne della matrice

Recupera il numero di righe o di colonne della matrice.

```
RigheMatrice(m)
ColonneMatrice(m)
```

dove, in entrambe le funzioni, *m* è la matrice. La prima restituisce il numero di righe della matrice, la seconda il numero di colonne.

5.3. Funzioni per la inizializzazione degli esempi

Matrici di esempio

Costruiscono matrici di esempio statiche, costanti, usate per testare le procedure di allineamento.

```
MatriceEsempio5x5()
MatriceEsempio21x21()
```

La prima costruisce una matrice di esempio con cinque righe e cinque colonne, la stessa usata in questo fascicolo per i primi esempi **(cfr. Un primo esempio a pagina 7)**, la seconda costruisce una matrice di 21 righe e 21 colonne che simula una distribuzione normale, anche questa presente nel fascicolo **(cfr 2.4 Il problema vero e proprio a pagina 10, la matrice è a pagina 11)**. Emtrambe restituiscono una matrice, implementata come una lista di righe (e ciascuna riga come una lista di elementi).

Funzione gaussiana e distribuzione del segnale

La prima funzione calcola il valore di una gaussiana bidimensionale:

```
Gaussiana(x, y)
```

dove *x, y* sono rispettivamente colonna e riga della posizione della matrice in cui calcolare il valore della funzione **(cfr. 2.4 Il problema vero e proprio a pagina 10)**. Fa uso delle variabili globali che abbiamo descritto all'inizio di questo capitolo. Restituisce un numero reale.

La seconda calcola un valore simile ma disturbato dall'errore:

```
GaussianaErrore(x, y)
```

dove *x, y* sono rispettivamente colonna e riga della posizione (nella matrice) in cui calcolare il valore **(cfr. Costruiamo la matrice a pagina 12)**. Usa *Gaussiana(x, y)*. Restituisce un numero

intero positivo: il valore del segnale moltiplicato per il fattore di scala.

Infine una funzione che costruisce la matrice che simula la distribuzione del segnale, disturbo compreso:

```
CreaMatriceTest(righe, colonne)
```

dove *righe, colonne* sono, ovviamente, il numero di righe e colonne della matrice da creare. Usa *GaussianaErrore(x, y)*. Restituisce una matrice come lista di righe, come di consueto.

5.4. Funzioni per la risoluzione del problema

Determinare la posizione vicina con il valore maggiore

Determina la casella della matrice vicina a quella attuale con il valore maggiore (in cui il segnale è migliore).

```
VicinaMaggiore(m, x, y)
```

dove *m* è la matrice e x, y rispettivamente colonna e riga della posizione (casella) attuale. Usa *ElementoMatrice(x, y)*. Restituisce una lista *[x_m, y_m]* contenente colonna e riga della casella vicina con il valore più alto.

Spostare il telescopio nella posizione migliore

Sposta il telescopio di casella in casella fino ad arrivare a quella in cui il valore del segnale è maggiore.

```
Allineamento(m, x_0, y_0)
```

dove *m* è la matrice presa in considerazione e x_0, y_0 colonna e riga della posizione di partenza. Usa *VicinaMaggiore(m, x, y)* e *ElementoMatrice(m, x, y)*. Restituisce il percorso fatto dal telescopio sottoforma di una lista in cui ogni elemento è una lista del tipo *[x, y]* dove *x, y* sono colonna e riga delle caselle attraversate (il primo elemento della lista è la casella di partenza, l'ultimo la casella di arrivo). Il percorso è quello effettivamente seguito dal telescopio, in particolare uno spostamento *in diagonale* è memorizzato come due spostamenti: il primo in verticale e il secondo in orizzontale.

Infine una funzione per convertire in una stringa di testo il percorso del telescopio:


```
PercorsoInTesto(percorso)
```

dove *percorso* è, come già detto, una lista del tipo *[x, y]* dove *x, y* sono colonna e riga delle caselle attraversate (il primo elemento della lista è la casella di partenza, l'ultimo la casella di arrivo). Restituisce una stringa di testo con l'elenco delle caselle attraversate.

5.5. Miglioramenti

La rilevazione dell'intensità del segnale è piuttosto laboriosa e richiede parecchio tempo, nell'ordine dei minuti per cui un miglioramento possibile, che abbiamo in programma ma che non abbiamo ancora implementato, cerca di ridurre per quanto possibile le misurazioni utilizzando valori già rilevati in precedenza (possiamo ipotizzare che le condizioni dell'atmosfera siano abbastanza stabili nel periodo di osservazione). Tenete conto che ogni volta che si esegue un'istruzione del tipo *M(..., ...)* nelle funzioni che abbiamo mostrato, c'è una misura dell'intensità del segnale.

La miglioria riguarda soprattutto la funzione *VicinaMaggiore* che crea una matrice ausiliaria,

chiamata nel seguito *M_2* e inizialmente *vuota*, in cui sono memorizzati i valori del segnale via via rilevati, questi valori sono usati successivamente per evitare di misurare più volte l'intensità del segnale nella stessa posizione.

```
VicinaMaggiore(x, y)
      Determina la casella, tra le otto vicine della casella specificata,
      quella che ha il valore più grande
      max = - inf
      x_m = x-1
      y_m = y-1
      se x-1 > 0 e y-1 > 0 allora
            casella diagonale in alto a sinistra
            se M_2(x-1, y-1) > -inf allora
                  confronta con il valore misurato in precedenza
                  se M_2(x-1, y-1) > max allora
                        max = M_2(x-1, y-1)
                        x_m = x-1
                        y_m = y-1
            altrimenti
                  rileva un valore non ancora misurato e lo salva
                  nella matrice ausiliaria
                  M_2(x-1,y-1) = M(x-1,y-1)
                  se M_2(x-1, y-1) > max allora
                        max = M_2(x-1, y-1)
                        x_m = x-1
                        y_m = y-1
      se y-1 > 0 allora
            casella adiacente sopra
            se M_2(x, y-1) > -inf allora
                  confronta con il valore misurato in precedenza
                  se M_2(x, y-1) > max allora
                        max = M_2(x, y-1)
                        x_m = x
                        y_m = y-1
            altrimenti
                  rileva un valore non ancora misurato e lo salva
                  nella matrice ausiliaria
                  M_2(x,y-1) = M(x,y-1)
                  se M_2(x, y-1) > max allora
                        max = M_2(x, y-1)
                        x_m = x
                        y_m = y-1
      se x+1 < colonne e y-1 > 0 allora
            casella diagonale in alto a destra
            se M_2(x+1, y-1) > -inf allora
                  ...
ritorna (x_m, y_m)
```

Per il resto l'applicazione rimane la stessa.

LICEO SCIENTIFICO GRASSI LATINA

CNR IAC
Istituto per le Applicazioni del Calcolo

CNR IFN
Istituto di Fotonica e Nanotecnologie

INM

ISMAR
Istituto di Scienze Marine

LSS G.B. Grassi
LICEO SCIENTIFICO STATALE G.B. GRASSI DI LATINA
WWW.LICEOGRASSILATINA.ORG

CNR - IAC
ISTITUTO PER LE APPLICAZIONI DEL CALCOLO MAURO PICONE
WWW.IAC.CNR.ORG

CNR - IFN Roma
ISTITUTO DI FOTONICA E NANOTECNOLOGIE
WWW.ROMA.IFN.CNR.ORG

CNR - INM
ISTITUTO DI INGEGNERIA DEL MARE
WWW.INSEAN.CNR.ORG

CNR - ISMAR
ISTITUTO DI SCIENZE MARINE
WWW.ISMAR.CNR.ORG