

Γεωμετρικοί μετασχηματισμοί εικόνας

Μάθημα: Υπολογιστική Οραση

Κ. Δελήμπασης

Γεωμετρικοί Μετασχηματισμοί

- Ορισμός σημείου στον Ευκλείδειο χώρο: $\mathbf{p}=[x_p, y_p, z_p]^T$, όπου x_p, y_p, z_p πραγματικοί αριθμοί.
- Εστω E^3 το σύνολο των $\mathbf{p}$.
- Ένας γεωμετρικός μετασχηματισμός $T(\boldsymbol{\pi})$, με διάνυσμα παραμέτρων $\boldsymbol{\pi}$, ορίζεται ως: $T: E^3 \rightarrow E^3$
- Οι μετασχηματισμοί σε 2D αποτελούν υποπερίπτωση των 3D.
- Παραδείγματα: Μεταφορά (translation), περιστροφή (rotation), αλλαγή κλίμακας (scaling).
- Κάθε γεωμετρικός μετασχηματισμός που μπορεί να περιγραφεί σαν συνδυασμός μεταφορών, περιστροφών, ή αλλαγής κλίμακας λέγεται γενικευμένος συσχετισμένος μετασχηματισμός (affine).

Ομογενείς συντεταγμένες

- Σύνθεση μετασχηματισμών
 - Ακολουθία μετασχηματισμών εκφράζεται σαν γινόμενο των αντίστοιχων πινάκων, με τον πίνακα του 1ου μετασχηματισμού αριστερά
 - Η μετατόπιση είναι ο μόνος μετασχηματισμός ο οποίος απαιτεί πρόσθεση
- Ομογενείς συντεταγμένες: για κάθε σημείο $P(x,y)$, εισάγουμε μία επιπλέον συντεταγμένη w , $P(x,y,w_0)$,
 - Το σημείο $P'(x/w_0, y/w_0, 1)$ αποτελεί την αναπαράσταση ομογενών συντεταγμένων στο επίπεδο $w=w_0$
 - Συνήθως χρησιμοποιείται η βασική αναπαράσταση με $w_0=1$
- Με την εφαρμογή των ομογενών συντεταγμένων η μετατόπιση γίνεται με πολλαπλασιασμό πινάκων και έτσι είναι δυνατή η σύνθεση πολλών διαδοχικών μετασχηματισμών σε ένα μόνο πίνακα μετασχηματισμού.

- Συχνά είναι απαραίτητο να εφαρμόσουμε ένα συσχετισμένο μετασχηματισμό σε μία εικόνα, πχ
 - Περιστροφή της εικόνας γύρω από το κέντρο μάζας της
 - Αλλαγή κλίμακας της εικόνας (ισοδύναμα interpolation).
- Εστω T ο μετασχηματισμός $T: E^3 \rightarrow E^3$ εικόνας που μετρέπει μία εικόνα I στην εικόνα I_1 .

$$I_1(T(x, y)) = I(x, y)$$

$$T(x, y) = \mathbf{T} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- $\mathbf{T}$ είναι ο πίνακας του μετασχηματισμού (σε ομογενείς συντεταγμένες), ο οποίος εφαρμοζόμενος στις συντεταγμένες ενός pixel παράγει τις συντεταγμένες του αντίστοιχου pixel της μετασχηματισμένης εικόνας.

- Μετατόπιση (Translation) κατά d_x, d_y :
- Αλλαγή κλίμακας (ανεξάρτητα σε κάθε διάσταση), ως προς την αρχή των αξόνων.
- Περιστροφή (Rotation) γύρω από τον άξονα των z , ως προς την αρχή των αξόνων.
- Οποιοσδήποτε συσχετισμένος μετασχηματισμός (στις 2 διαστάσεις) μπορεί να περιγραφεί από ένα πίνακα της μορφής:

$$T = \begin{pmatrix} 1 & 0 & d_x \\ 0 & 1 & d_y \\ 0 & 0 & 1 \end{pmatrix}$$

$$T(s_x, s_y) = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$R(\theta) = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{T} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix}$$

- Ο μετασχηματισμός Affine εφαρμόζεται ως εξής:
 - Για κάθε pixel (x_1, y_1) της I_1 , υπολογίζονται οι νέες συντεταγμένες (x_2, y_2) ως εξής:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

- Επισημαίνεται ότι:
 - Τα (x_2, y_2) δεν είναι πάντα ακέραια, μπορεί να πάρουν και αρνητικές τιμές, ή τιμές μεγαλύτερες από τον αριθμό γραμμών και στηλών της I_1 .
 - Εφαρμογή του μετασχηματισμού σε Matlab:
 - `I2=imtransform(I1,T,'nearest','XData',XData,'YData',YData,'FillValues',0);`

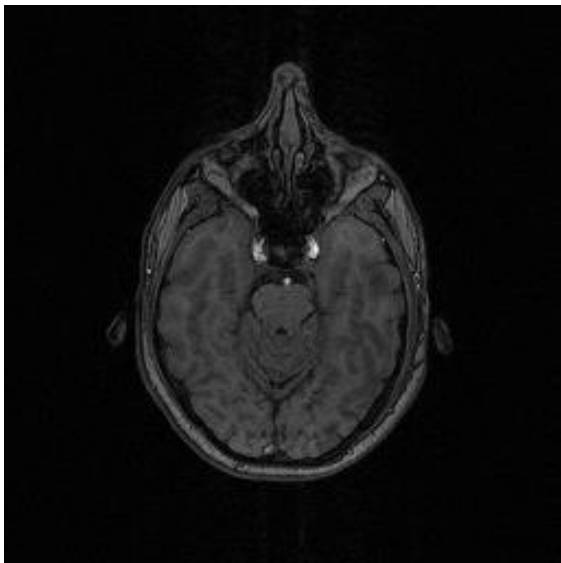
Παράδειγμα

- Εστω ότι επιθυμούμε να μετατοπίσουμε μία εικόνα μεγέθους 256x256 κατά 10,-8 στους δύο άξονες, να την περιστρέψουμε κατά $\vartheta=6^\circ$ και να αλλάξουμε την κλίμακα στον άξονα X και Y κατά 1.2 και 0.8 αντίστοιχα.

$$T = \underbrace{\begin{bmatrix} 1 & 0 & c_x \\ 0 & 1 & c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας μετατόπισης (0,0) --> κέντρο μάζας}} \underbrace{\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας αλλαγής κλίμακας}} \underbrace{\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας περιστροφής}} \underbrace{\begin{bmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας μετατόπισης dx, dy}} \underbrace{\begin{bmatrix} 1 & 0 & -c_x \\ 0 & 1 & -c_y \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας μετατόπισης κέντρου μάζας --> (0,0)}}$$

$$T = \underbrace{\begin{bmatrix} 1 & 0 & 128 \\ 0 & 1 & 128 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας μετατόπισης (0,0) --> κέντρο μάζας}} \underbrace{\begin{bmatrix} 1.2 & 0 & 0 \\ 0 & 0.8 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας αλλαγής κλίμακας}} \underbrace{\begin{bmatrix} \cos\left(\frac{6\pi}{180}\right) & -\sin\left(\frac{6\pi}{180}\right) & 0 \\ \sin\left(\frac{6\pi}{180}\right) & \cos\left(\frac{6\pi}{180}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας περιστροφής +6 μοιρών}} \underbrace{\begin{bmatrix} 1 & 0 & 10 \\ 0 & 1 & -8 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας μετατόπισης dx, dy}} \underbrace{\begin{bmatrix} 1 & 0 & -128 \\ 0 & 1 & -128 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{Πίνακας μετατόπισης κέντρου μάζας --> (0,0)}}$$

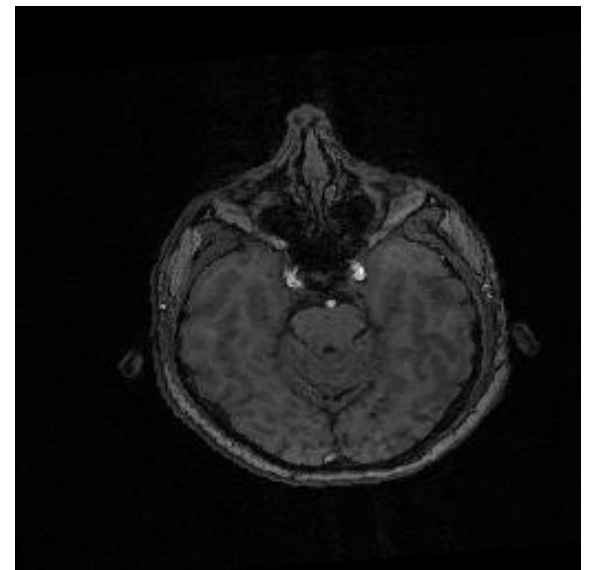
- Εφαρμογή του μετασχηματισμού του προηγούμενου παραδείγματος



Αρχική εικόνα

$$T = \begin{bmatrix} 1.1934 & -0.1254 & -8.7030 \\ 0.0836 & 0.7956 & 15.4572 \\ 0 & 0 & 1 \end{bmatrix}$$

Πίνακας
Μετασχηματισμού



Μετασχηματισμένη
εικόνα

Απλός αλγόριθμος εφαρμογής συσχετισμένου μετασχηματισμού: μη ενδεικνυόμενος

Εστω ότι θέλουμε να μετασχηματίσουμε γεωμετρικά δοθείσα εικόνα I_1 , βάσει συσχετισμένου μετασχηματισμού με πίνακα A .

Βήμα 1ο: Προσδιορίζουμε τη προβολή (απεικόνιση) του κέντρου του κάθε pixel (x,y) της αρχικής εικόνας I_1 στη νέα εικόνα I_2 :

$$(x_1, y_1, 1)^T = A \cdot (x, y, 1)^T.$$

Βήμα 2ο: Βρίσκουμε το pixel της νέας εικόνας I_2 του οποίου το κέντρο βρίσκεται πιο κοντά στο σημείο της προβολής (στρογγυλοποίηση συντεταγμένων)

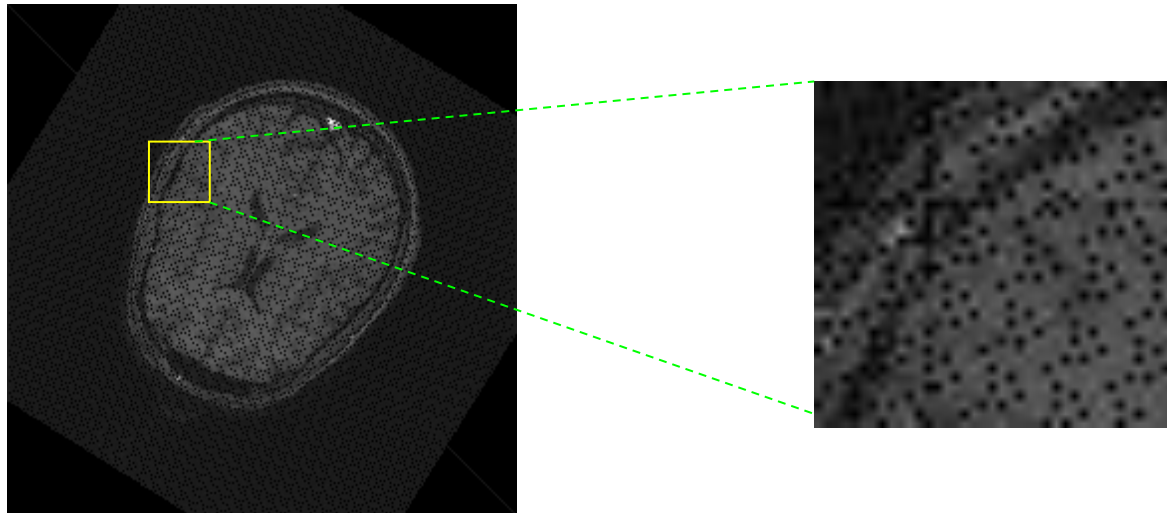
Βήμα 3ο: Θέτουμε τη φωτεινότητα του ζητούμενου pixel της τελικής εικόνας ίση με τη φωτεινότητα του pixel, από το βήμα 2, της αρχικής εικόνας:

$$I_2(\text{round}(x_1), \text{round}(y_1)) = I_1(x, y) .$$

Παράδειγμα ψευδοκώδικα του προηγούμενου αλγόριθμου για περιστροφή εικόνας κατά γωνία θ γύρω από το κέντρο μάζας της CM.

```
A=T(-CM) R(θ) T(CM)
for i=1:256
    for j=1:256
        [i1,j1]T=A*[i,j]T
        if i1>256 →i1=256;
        if i1<=1→i1=1;
        if j1>256 →j1=256;
        if j1<=1→j1=1;
        IM(round(i1),round(j1))=I(i,j);
    end;
end;
```

Παράδειγμα περιστροφής εικόνας με χρήση του προηγούμενου αλγόριθμου



Παρατηρούμε το artifact των μηδενικών Pixel (τα οποία δεν έχουν πάρει τιμές). Η χρήση της παρεμβολής βάσει του κοντινότερου γείτονα είναι πολύ απλή στην υλοποίηση αλλά έχει το μειονέκτημα ότι θολώνει την περιστραμμένη εικόνα. Εναλλακτικά μπορεί να χρησιμοποιηθεί η διγραμμική παρεμβολή.

Γενικός αλγόριθμος εφαρμογής συσχετισμένου μετασχηματισμού

- *Βήμα 1ο:* Προσδιορίζουμε τη προβολή (απεικόνιση) του κέντρου του κάθε pixel (x,y) της νέας εικόνας I_2 στην αρχική εικόνα I_1 :

$$(x_1, y_1, 1)^T = A^{-1} \cdot (x, y, 1)^T.$$

Υπολογίζουμε την τιμή του (x,y) της νέας εικόνας I_2 με 2 τρόπους:

- *Βήμα 2α: Παρεμβολή κοντινότερου γείτονα:* Βρίσκουμε το pixel της αρχικής εικόνας του οποίου το κέντρο βρίσκεται πιο κοντά στο σημείο της προβολής με στρογγυλοποίηση:

$$I_2(x,y) = I_1(\text{round}(x_1), \text{round}(y_1))$$

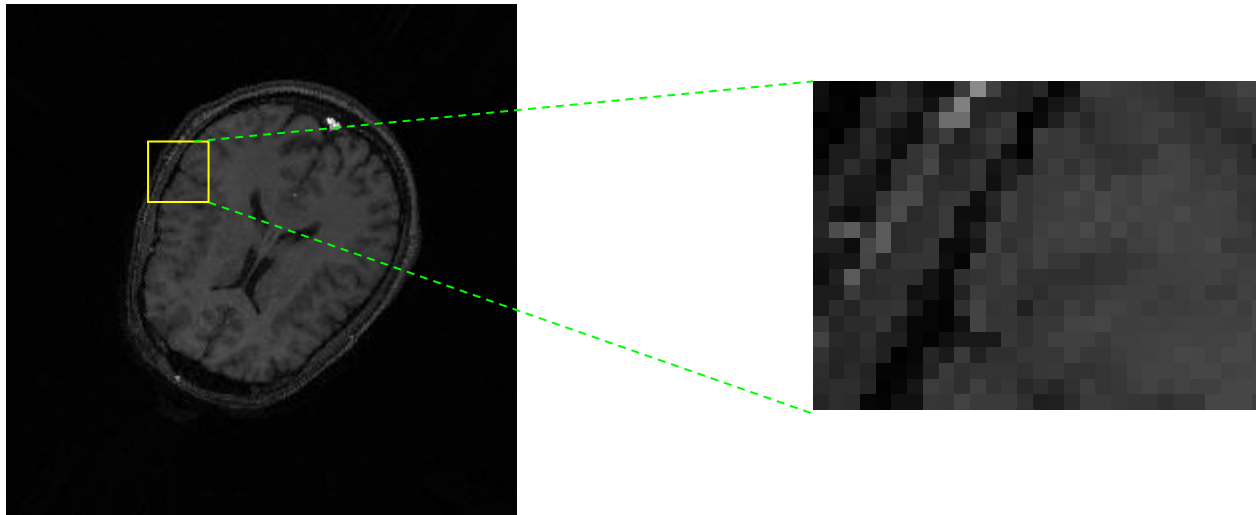
- *Βήμα 2β: Διγραμμική Παρεμβολή:* Θέτουμε την τιμή του (x_1, y_1) ζητούμενου pixel της τελικής εικόνας I_2 , ίση με την τιμή της αρχικής εικόνας I_1 στη θέση (x,y) . Επειδή τα (x_1, y_1) είναι εν γένει πραγματικοί αριθμοί, εφαρμόζουμε διγραμμική παρεμβολή:

$$\begin{aligned} I_2(y, x) &= I_1(y_1, x_1) = I_1([y_1] + a, [y_1] + b) = \\ &= (1-a)(1-b)I([y_1], [x_1]) + a(1-b)I([y_1], [x_1] + 1) + (1-a)bI_1([y_1] + 1, [x_1]) \\ &+ abI_1([y_1] + 1, [x_1] + 1) \end{aligned}$$

Αλγόριθμος περιστροφής εικόνας για διόρθωση artifact, βάσει του κοντινότερου γείτονα

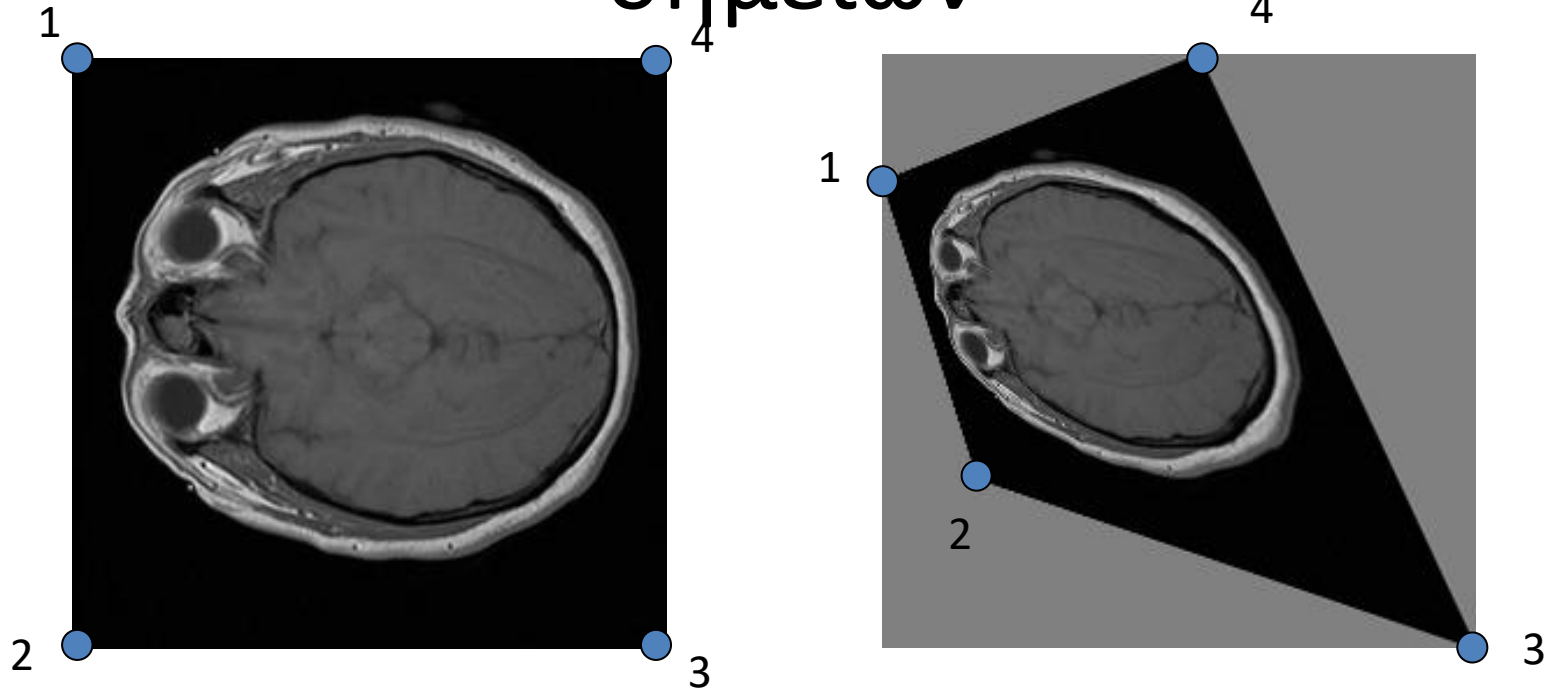
```
A=T (-CM) R (θ) T (CM)           // Υπολογισμός του πίνακα περιστροφής A
for i=1:256                         // Για κάθε pixel της νέας εικόνας I2
    for j=1:256
        [i1,j1]T=(A-1) * [i,j]T    // Υπολογισμός του pixel της I1 από το οποίο
        // προέρχεται με χρήση του αντίστροφου του A
        if i1>256 →i1=256;
        if i1<=1→i1=1;
        if j1>256 →j1=256;          // Έλεγχος αν το (i1,j1) είναι εντός
        // της αρχικής εικόνας
        if j1<=1→j1=1;
        IM2(i,j)=
            IM(round(i1), round(j1)) ; // Μέθοδος του κοντινότερου γείτονα
    end;
end;
```

Παράδειγμα περιστροφής εικόνας χωρίς artifacts με χρήση του προηγούμενου αλγόριθμου



Παρατηρούμε ότι το artifact των μηδενικών Pixel έχει διορθωθεί. Η χρήση της παρεμβολής βάσει του κοντινότερου γείτονα είναι πολύ απλή στην υλοποίηση αλλά έχει το μειονέκτημα ότι θολώνει την περιστραμμένη εικόνα. Εναλλακτικά μπορεί να χρησιμοποιηθεί η διγραμμική παρεμβολή.

Παραμόρφωση εικόνας με χρήση 4 σημείων



- Μετασχηματισμοί όπως διγραμμικός και προβολικός απαιτούν τουλάχιστον 4 ζεύγη ομόλογων σημείων

- Γενική μορφή μετασχηματισμού

$$\begin{aligned}x_1 &= F_1(x, y) \\y_1 &= F_2(x, y)\end{aligned}$$
- Διγραμμικός (δεν διατηρεί ευθείες)

$$\begin{aligned}x_1 &= a_0 + a_1x + a_2y + a_3xy \\y_1 &= b_0 + b_1x + b_2y + b_3xy\end{aligned}$$
- Προβολικός (διατηρεί ευθείες)

$$x_1 = \frac{a_0 + a_1x + a_2y}{1 + c_1x + c_2y}, y_1 = \frac{b_0 + b_1x + b_2y}{1 + c_1x + c_2y}$$
- Οι παράμετροι του μετασχηματισμού υπολογίζονται από τα 4 σημεία με τις μετατοπίσεις τους

Γεωμετρικοί μετασχηματισμοί ομογενών συντεταγμένων σε 3 διαστάσεις (3D)

- Μετατόπιση,

$$\mathbf{T}(\mathbf{d}) = \begin{pmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Αντίστροφος μετασχηματισμός

$$\mathbf{T}^{-1}(\mathbf{d}) = \mathbf{T}(-\mathbf{d})$$

- Αλλαγή κλίμακας:

$$\mathbf{S} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Αντίστροφος μετασχηματισμός

$$\mathbf{S}^{-1}(s_x, s_y, s_z) = \mathbf{S}(s_x^{-1}, s_y^{-1}, s_z^{-1})$$

Περιστροφή γύρω από τους 3 άξονες: Γωνίες Euler

- Πίνακες περιστροφής γύρω από τους άξονες X , Y , Z κατά γωνία θ . Ομογενείς συντεταγμένες.
- Αντίστροφος μετασχηματισμός $R_x(-\theta_x)$
- Προφανώς $R_x(\theta_x) R_x(-\theta_x) = I$

$$\mathbf{R}_x = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta_x) & -\sin(\theta_x) & 0 \\ 0 & \sin(\theta_x) & \cos(\theta_x) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{R}_y = \begin{pmatrix} \cos(\theta_y) & 0 & -\sin(\theta_y) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\theta_y) & 0 & \cos(\theta_y) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\mathbf{R}_z = \begin{pmatrix} \cos(\theta_z) & -\sin(\theta_z) & 0 & 0 \\ \sin(\theta_z) & \cos(\theta_z) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Γενική μορφή affine 3D με ομογενείς συντεταγμένες

Μετασχημ. σημείο

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Συντεταγμ. σημείου

Πίνακας Affine Μετασχ.

- Για να μετασχηματίσουμε N σημεία:

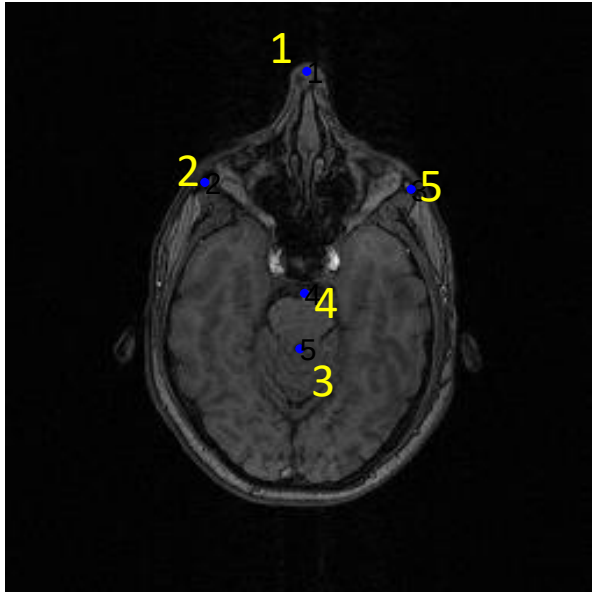
$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \dots & x_N \\ y_1 & y_2 & \dots & y_N \\ z_1 & z_2 & \dots & z_N \\ 1 & 1 & \dots & 1 \end{pmatrix}$$

Χωρική ταύτιση εικόνων:

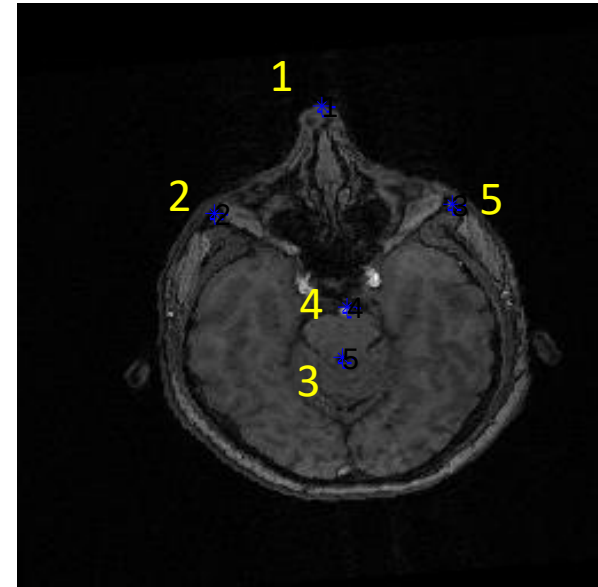
Καθορισμός του πίνακα μετασχηματισμού βάσει ομόλογων σημείων

- Εστω 2 εικόνες I_1, I_2 του ίδιου αντικειμένου, που έχουν συλλεχθεί υπό διαφορετική γεωμετρία.
- Εστω ένας αριθμός από ζεύγη **ομόλογων** σημείων μεταξύ δύο εικόνων: $\{p_i^A\}$ στην I_1 και $\{p_i^B\}$ στην I_2 .
- Ζητείται ο πίνακας που μετασχηματίζει γεωμετρικά την I_1 στην I_2 , ώστε τα μετασχηματισμένα σημεία $\{p_i^A\}$ να συμπίπτουν με τα $\{p_i^B\}$.
 - Λέμε τότε ότι οι δύο εικόνες ταυτίζονται χωρικά (spatial registration).
- Τα ζεύγη ομόλογων σημείων $\{p_i^A\}$ στην I_1 και $\{p_i^B\}$ στην I_2 , $i=1,\dots,N$ $N>3$, ορίζονται είτε από το χρήστη είτε από κάποια αυτόματη μέθοδο.
- **Ομόλογα** είναι δύο σημεία πάνω στα ίδια αντικείμενα στις δύο διαφορετικές εικόνες.
 - Οι εικόνες δεν ταυτίζονται χωρικά, άρα οι συντεταγμένες δύο ομόλογων σημείων δεν θα είναι ίδιες (πχ η μύτη του ασθενή στην I_1 δεν βρίσκεται στα pixel στα οποία βρίσκεται η μύτη του ίδιου ασθενή στην I_2).

Image I1



Transformed I1



- Παράδειγμα δύο εικόνων I1 και I2 του ίδιου αντικειμένου (MRI εγκεφάλου) με 5 ζεύγη ομολόγων σημείων που έχουν τοποθετηθεί σε κοινές ανατομικές δομές από τον χρήστη.

- Για να καθορίσουμε τον μετασχηματισμό Affine χρειαζόμαστε τον πίνακα του μετασχηματισμού ο οποίος έχει 6 αγνώστους:

$$a_{00}, a_{01}, a_{02}, a_{10}, a_{11}, a_{12}$$

- Οι 3 άγνωστοι a_{00}, a_{01}, a_{02} υπολογίζονται από τις X συντεταγμένες των ζευγών ομολόγων σημείων και
- οι 3 άγνωστοι a_{10}, a_{11}, a_{12} υπολογίζονται από τις Y συντεταγμένες.
- Αν $N > 3$ (συνήθης περίπτωση) τότε τα 2 γραμμικά συστήματα είναι υπερκαθορισμένα.

$$A = \begin{bmatrix} x_1^A & y_1^A & 1 \\ x_2^A & y_2^A & 1 \\ \dots & \dots & \dots \\ x_N^A & y_2^A & 1 \end{bmatrix}$$

- Εστω οι πίνακες

$$p = (a_{00}, a_{01}, a_{02})^T, q = (a_{10}, a_{11}, a_{12})^T$$

$$\mathbf{b}_1 = (x_1^A, x_2^A, \dots, x_N^A)^T, \mathbf{b}_2 = (x_1^B, x_2^B, \dots, x_N^B)^T$$

- Πρέπει να επιλυθούν τα γραμμικά συστήματα

$$\mathbf{A}\mathbf{p} = \mathbf{b}_1$$

$$\mathbf{A}\mathbf{q} = \mathbf{b}_2$$

- Ο A είναι διαστάσεων $N \times 3$ ενώ τα $\mathbf{p}, \mathbf{q}$ είναι διαστάσεων 3×1 και τα $\mathbf{b}_1, \mathbf{b}_2$ είναι διαστάσεων $N \times 1$.

– Μπορεί να χρησιμοποιηθεί ο τελεστής «\» του Matlab: $p = A \backslash b_1$ και $q = A \backslash b_2$.

- Η παραπάνω λύση ισοδυναμεί με τη μέθοδο ελαχίστων τετραγώνων:

$$(\mathbf{A}^T \mathbf{A})\mathbf{p} = \mathbf{A}^T \mathbf{b}_1$$

$$(\mathbf{A}^T \mathbf{A})\mathbf{q} = \mathbf{A}^T \mathbf{b}_2$$

Projective transformations

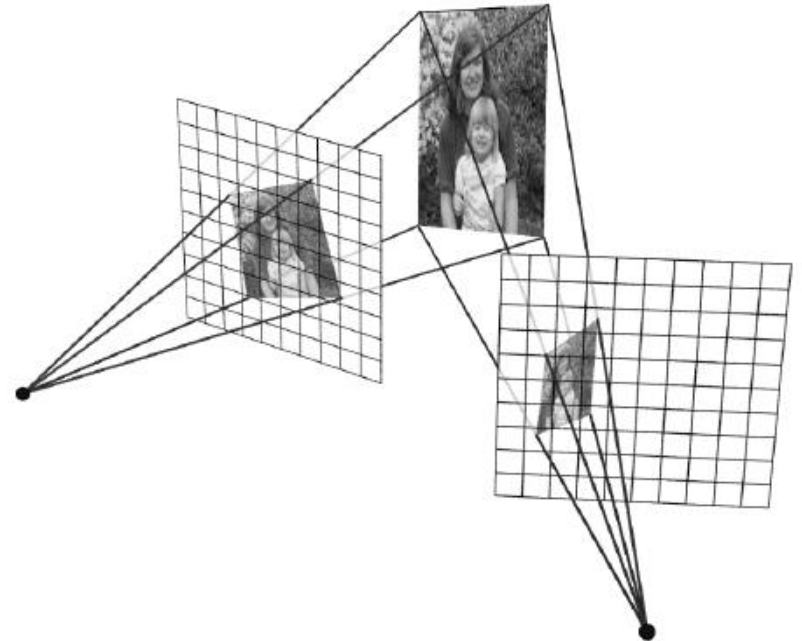
Projective transformations are combos of

- Affine transformations, and
- Projective warps

Properties of projective transformations:

- Lines map to lines
- Parallel lines do not necessarily remain parallel
- Ratios are preserved
- Closed under composition
- Models change of basis
- Projective matrix is defined up to a scale (8 DOF)

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

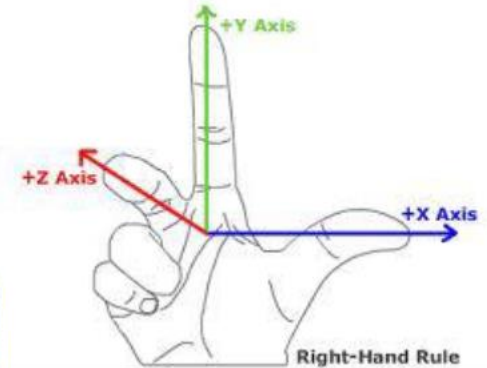
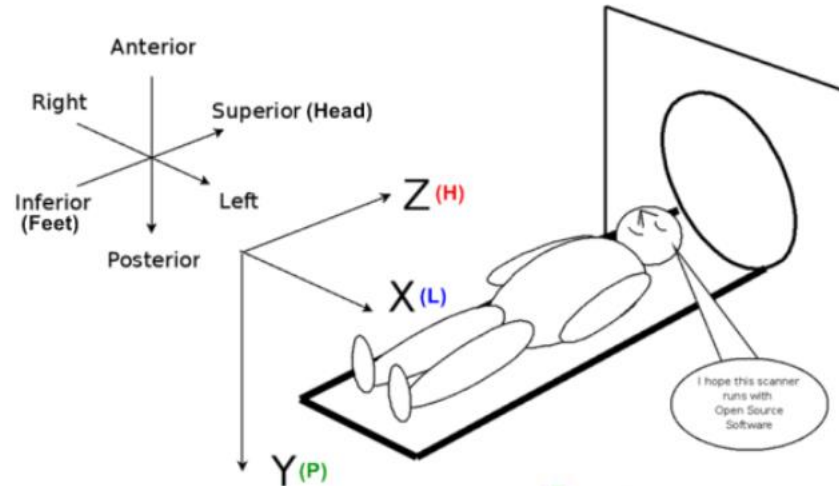


Geometric information in DICOM images

- Image Position (0020,0032) specifies the x, y, and z coordinates of the upper left hand corner of the image; it is the center of the first voxel transmitted.
- Image Orientation (0020,0037) specifies the direction cosines of the first row and the first column with respect to the patient

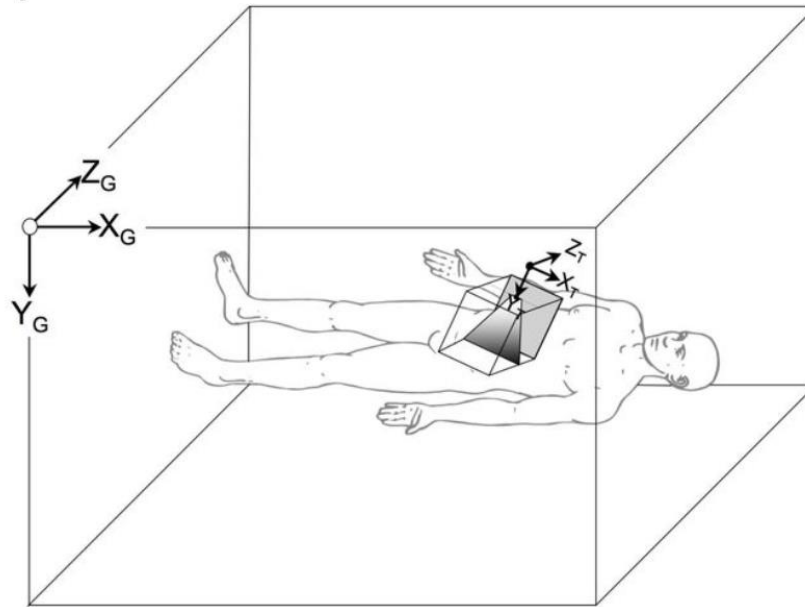
Attribute	Tag	Type	Attribute Description
Pixel Spacing	(0028,0030)	1	Physical distance in the patient between the center of each pixel, specified by a numeric pair - adjacent row spacing (delimiter) adjacent column spacing in <i>mm</i> . (Pixel Spacing = Row Spacing \ Colum Spacing = 0.30\0.25)
Image Orientation (Patient)	(0020,0037)	1	The direction cosines of the first row and the first column with respect to the patient.
Image Position (Patient)	(0020,0032)	1	The x, y, and z coordinates of the upper left hand corner (center of the first voxel transmitted) of the image, in <i>mm</i> .
Slice Thickness	(0018,0050)	2	Nominal slice thickness, in <i>mm</i> .
Slice Location	(0020,1041)	3	Relative position of exposure expressed in <i>mm</i> .

- DICOM uses a right handed LPH (=LPS) coordinate system.
- Relates to patient, not scanner.
- DICOM provides public fields that relate a 2D image to 3D patient space.
 - ImagePositionPatient
 - Coordinate of top left image pixel.
 - ImageOrientationPatient
 - Unit vector along an image row, the unit vector along a column.



<http://mrl.cs.uh.edu/matlab/matlab2.pdf>

Patient Frame Of Reference

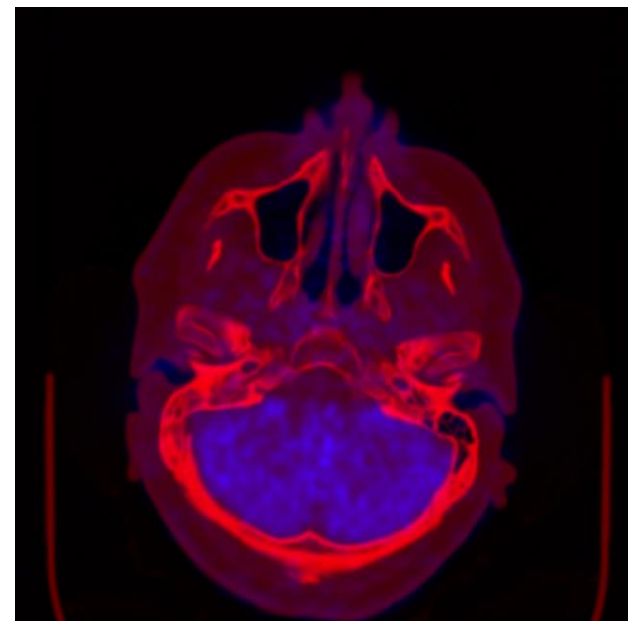
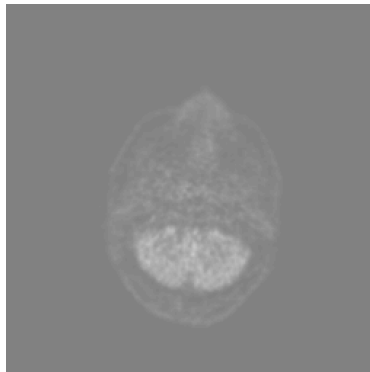
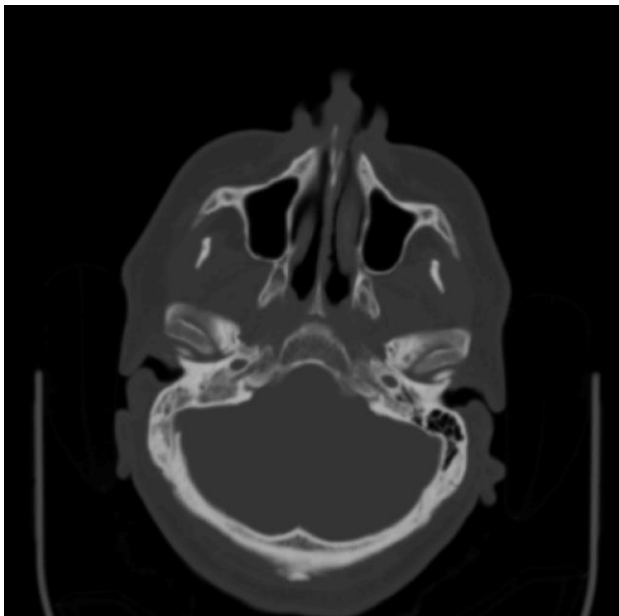


- The concept of direction cosines

Geometric information in DICOM images

$$\begin{bmatrix} P_x \\ P_y \\ P_z \\ 1 \end{bmatrix} = \begin{bmatrix} X_x \Delta i & Y_x \Delta j & 0 & S_x \\ X_y \Delta i & Y_y \Delta j & 0 & S_y \\ X_z \Delta i & Y_z \Delta j & 0 & S_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i \\ j \\ 0 \\ 1 \end{bmatrix} = M \begin{bmatrix} i \\ j \\ 0 \\ 1 \end{bmatrix}$$

- P_{xyz} : The coordinates of voxel (i,j) in the frame's image plane (mm).
- S_{xyz} : The the location in mm from the origin of the RCS (Image Position Patient) (0020,0032).
- X_{xyz} : The values from the row (X) direction cosine of the Image Orientation (Patient) (0020,0037) attribute.
- Y_{xyz} : The values from the column (Y) direction cosine of the Image Orientation (Patient) (0020,0037).
- i, j : Column index to the image plane (starts from 0).
- $\Delta i \Delta j$: Column Row pixel resolution of the Pixel Spacing (0028,0030) attribute in units of mm.



CT, PET, fused

Ελαστικοί γεωμετρικοί μετασχηματισμοί

- Πολλά είδη μετασχηματισμών δε προσεγγίζονται από ολικούς (global) μετασχηματισμούς, όπως affine, bilinear, projective κλπ
- Ελαστική παραμόρφωση λόγω
 - Αναπνοής, κίνησης καρδιάς, μεταβολή σχήματος στομάχου, ουροδόχος κύστη κλπ
- Ο πιο γνωστός ελαστικός γεωμετρικός μετασχηματισμός είναι το μοντέλο TPS (Thin Plate Splines) (Bookstein 1989).

Το μοντέλο TPS (Thin Plate Splines)

- Εστω ότι έχουμε επιλέξει 2 σύνολα ομόλογων σημείων $\{x_i, y_i\}, \{x'_i, y'_i\}$
- Κατασκευάζουμε τους πίνακες P, Y

$$P = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ \dots & \dots & \dots \\ 1 & x_n & y_n \end{bmatrix}, 3 \times n; \quad V = \begin{bmatrix} x'_1 & x'_2 & \dots & x'_n \\ y'_1 & y'_2 & \dots & y'_n \end{bmatrix}$$

- Ορίζουμε την συνάρτηση $U(r)=r^2 \log r^2 : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ και κατασκευάζουμε τον πίνακα K , όπου r_{ij} η απόσταση των σημείων i, j .

$$K = \begin{bmatrix} 0 & U(r_{12}) & \dots & U(r_{1n}) \\ U(r_{21}) & 0 & \dots & U(r_{2n}) \\ \dots & \dots & \dots & \dots \\ U(r_{n1}) & U(r_{n2}) & \dots & 0 \end{bmatrix}, n \times n;$$

- Κατασκευάζουμε τον πίνακα L

$$L = \left[\begin{array}{c|c} K & P \\ \hline P^T & O \end{array} \right], (n + 3) \times (n + 3)$$

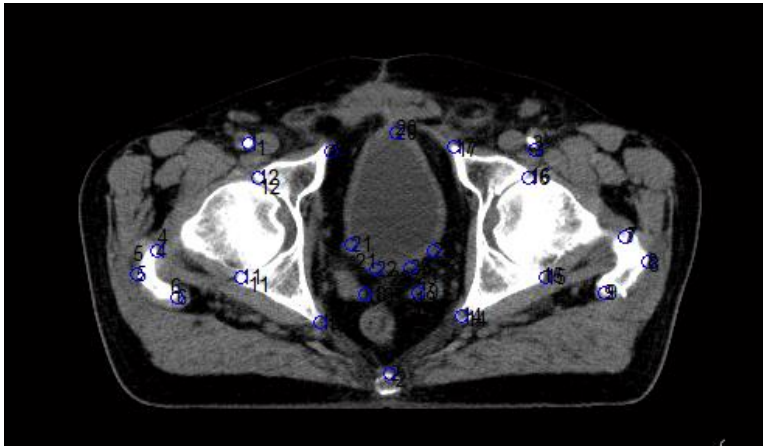
- Κατασκευάζουμε τον πίνακα $Y = (V | 0 \ 0 \ 0)^T$

$$L^{-1}Y = (W | a_1 \ a_x \ a_y)^T$$

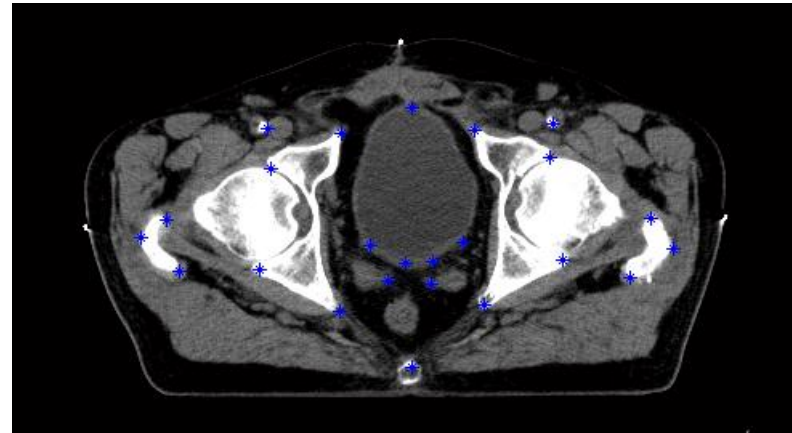
- Υπολογίζουμε για κάθε σημείο (x,y) τις νέες του συντεταγμένες βάσει της ακόλουθης διανυσματικής συνάρτησης:

$$f(x, y) = a_1 + a_x x + a_y y + \sum_{i=1}^n w_i U(|P_i - (x, y)|)$$

Παράδειγμα: ταύτιση εικόνων με ελαστικό μετασχηματισμό

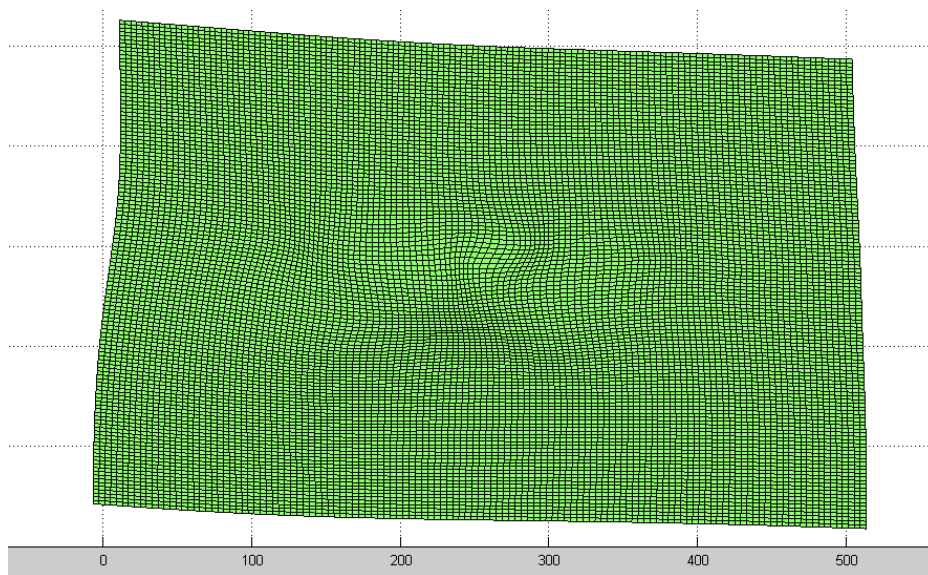


(α)

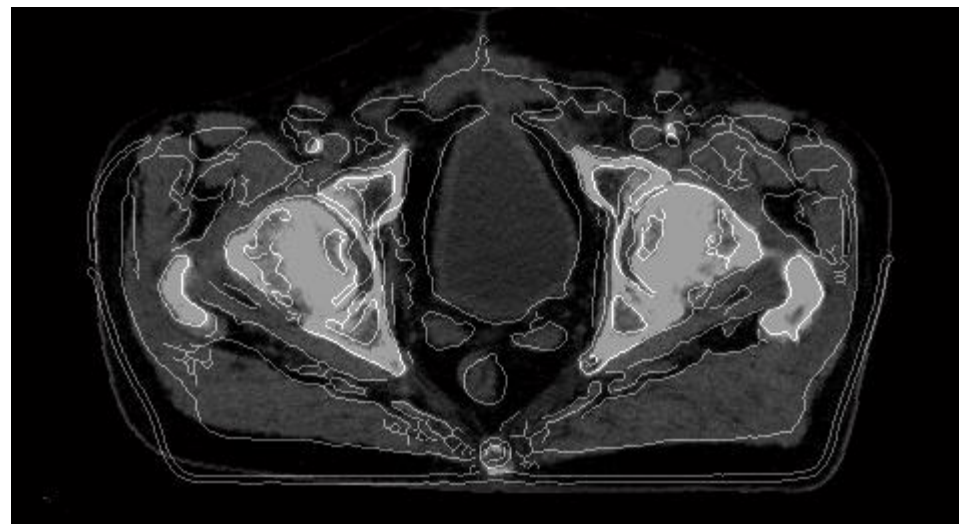


(β)

- Εστω δύο εικόνες του ίδιου αντικειμένου που έχει παραμορφωθεί ελαστικά.
- Ορίζουμε ζεύγη ομόλογων σημείων στις εικόνες
- Αναζητούμε τον μετασχηματισμό που παραμορφώνει τη (α) ώστε να ταυτιστεί με την (β)



Η εφαρμογή του ελαστικού
μετασχηματισμού σε συνθετική
εικόνα



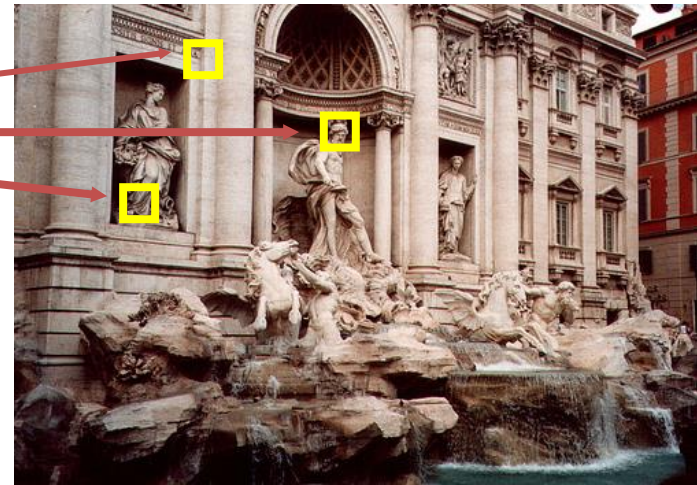
Οι ακμές της I2 μετά την εφαρμογή του ελαστικού
μετασχηματισμού προβεβλημένες επί της I1

Feature matching

- Given a feature in I_1 , how to find the best match in I_2 ?
 1. Define distance function that compares two descriptors.
 2. Test all the features in I_2 , find the one with min distance.



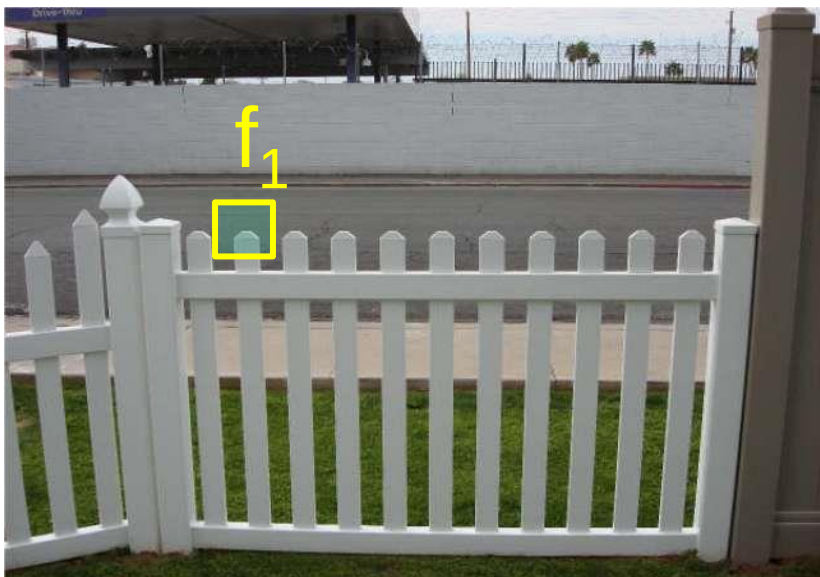
I_1



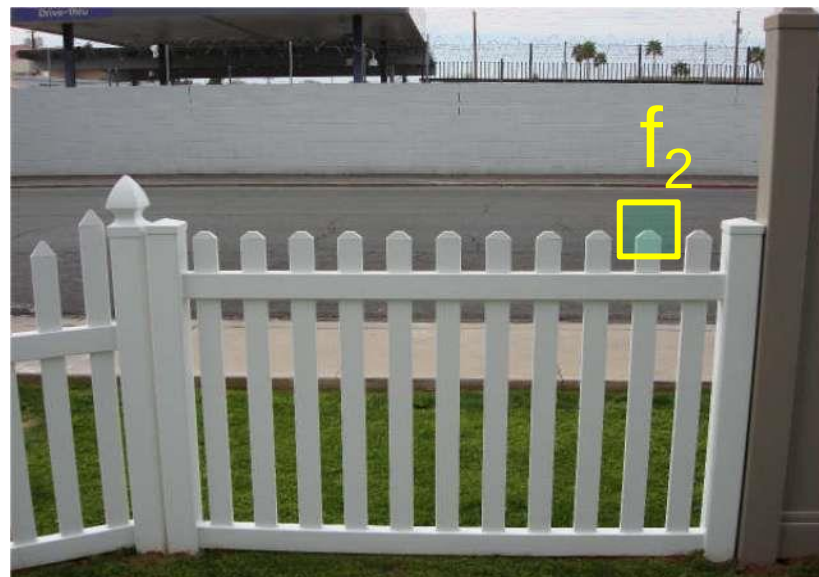
I_2

Feature matching: simple Euclidean distance may produce erroneous results

$$SSD = \sqrt{\sum_{i=1}^N (f_i - g_i)^2}$$

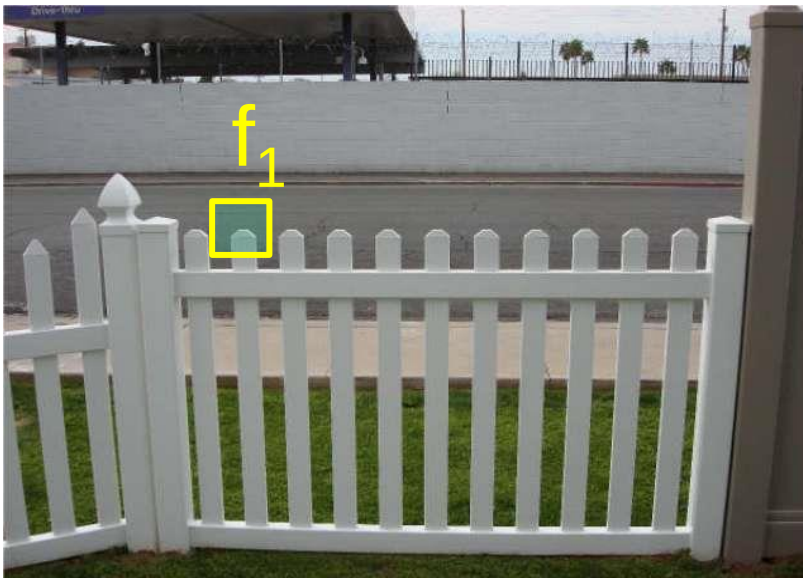


I_1



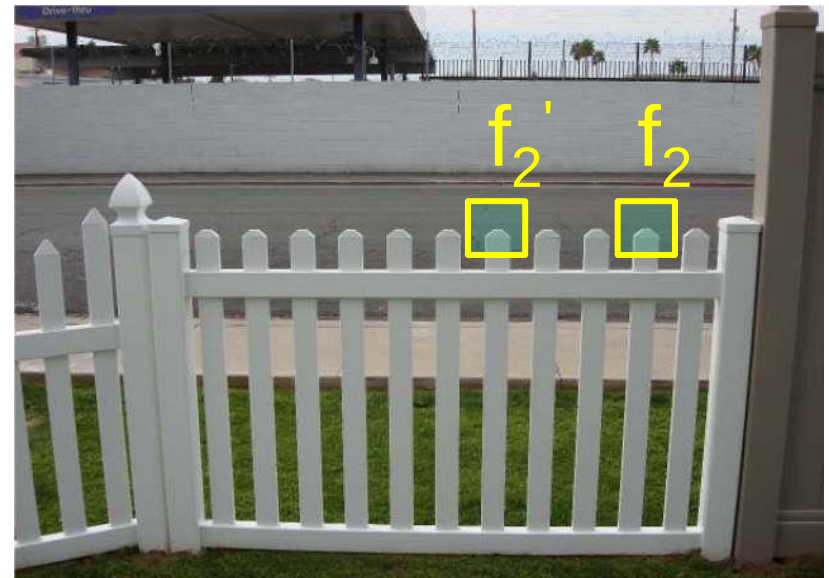
I_2

- A better distance measure is the following:
 - $\text{SSD}(f_1, f_2) / \text{SSD}(f_1, f_2')$
 - f_2 is best SSD match to f_1 in I_2
 - f_2' is 2nd best SSD match to f_1 in I_2



I_1

Κ. Δελήμπας

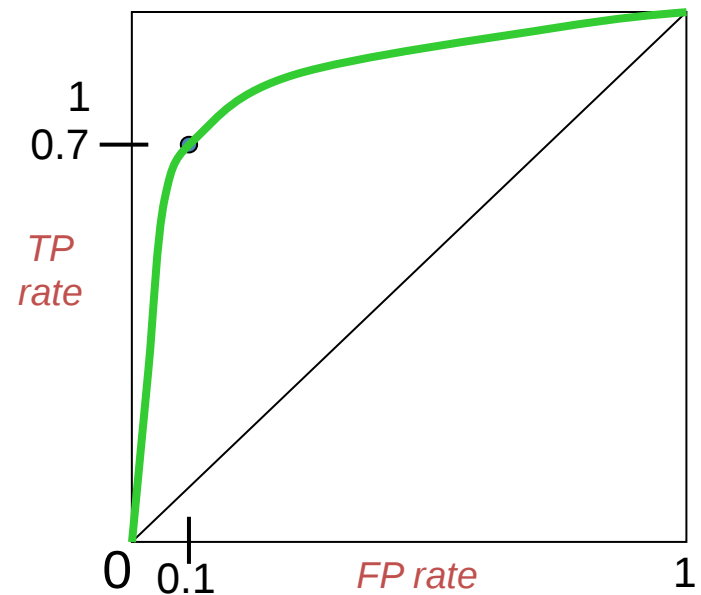


I_2

- ROC Curve

- Generated by computing (FP, TP) for different thresholds.

- Maximize area under the curve (AUC).

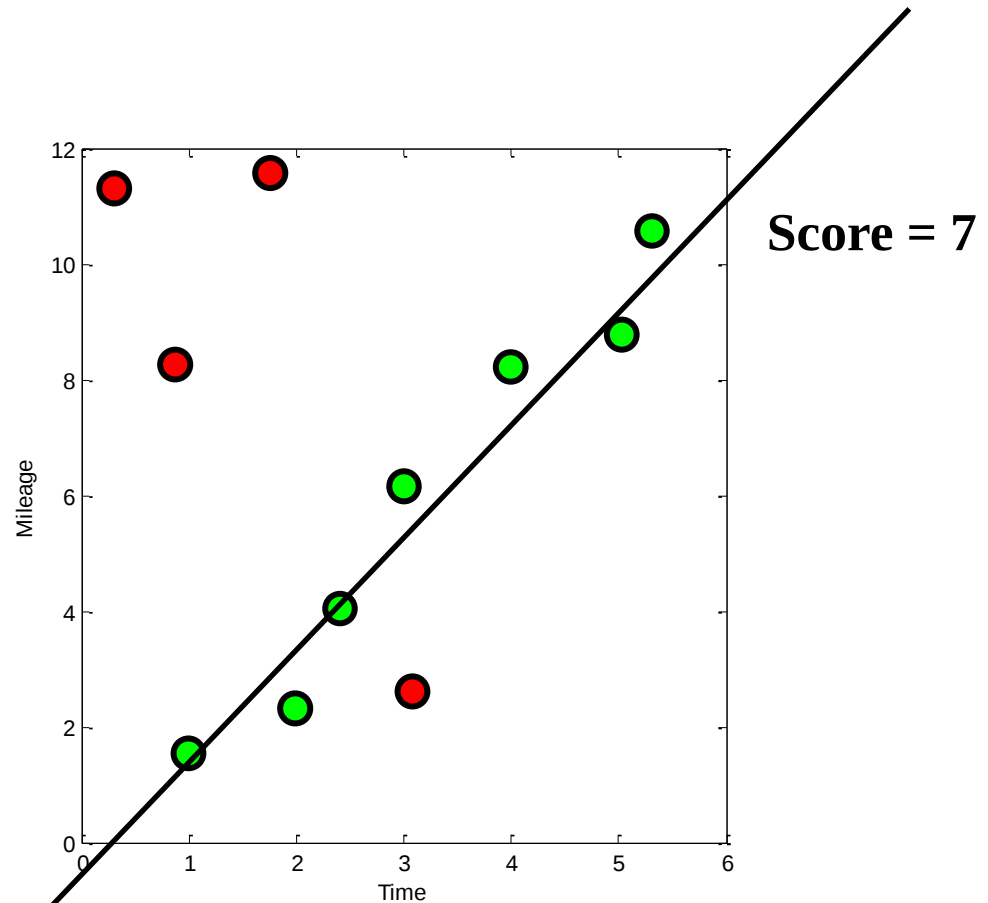
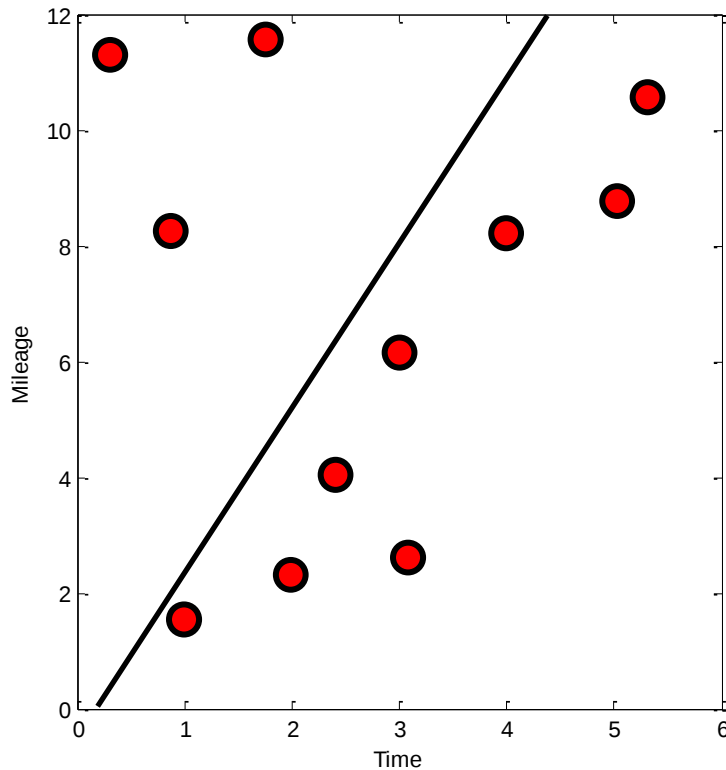


http://en.wikipedia.org/wiki/Receiver_operating_characteristic

RANSAC

- A more robust approach for parameter estimation of geometric transformation
 - Immune to outliers
 - Can be applied to many geometric models
 - Easy to code
 - Computationally efficient

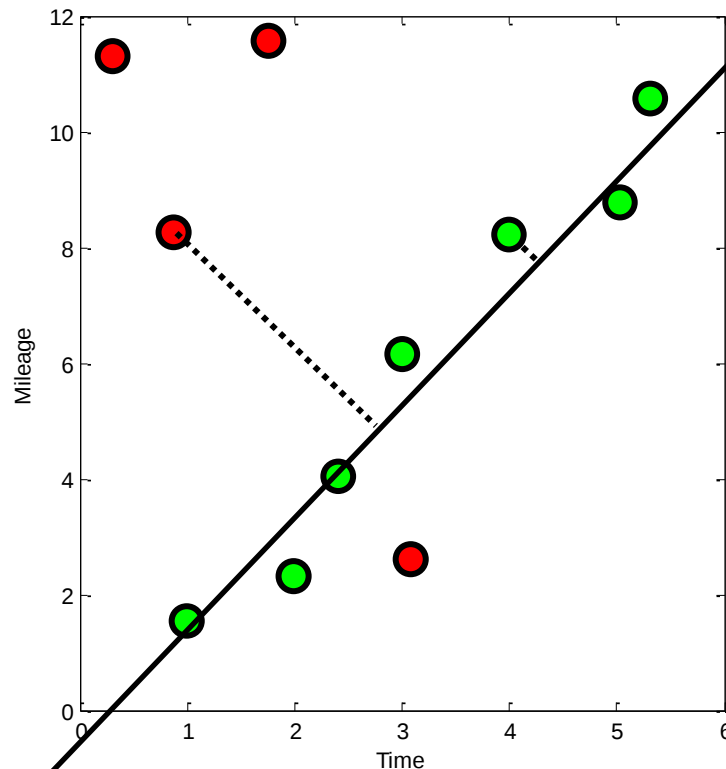
Linear regression vs Robust approaches



<http://www.cs.cornell.edu/courses/cs1114>, Prof. Noah Snaveley

Testing goodness

- How can we tell if a point agrees with a line?
- Compute the distance the point and the line, and threshold



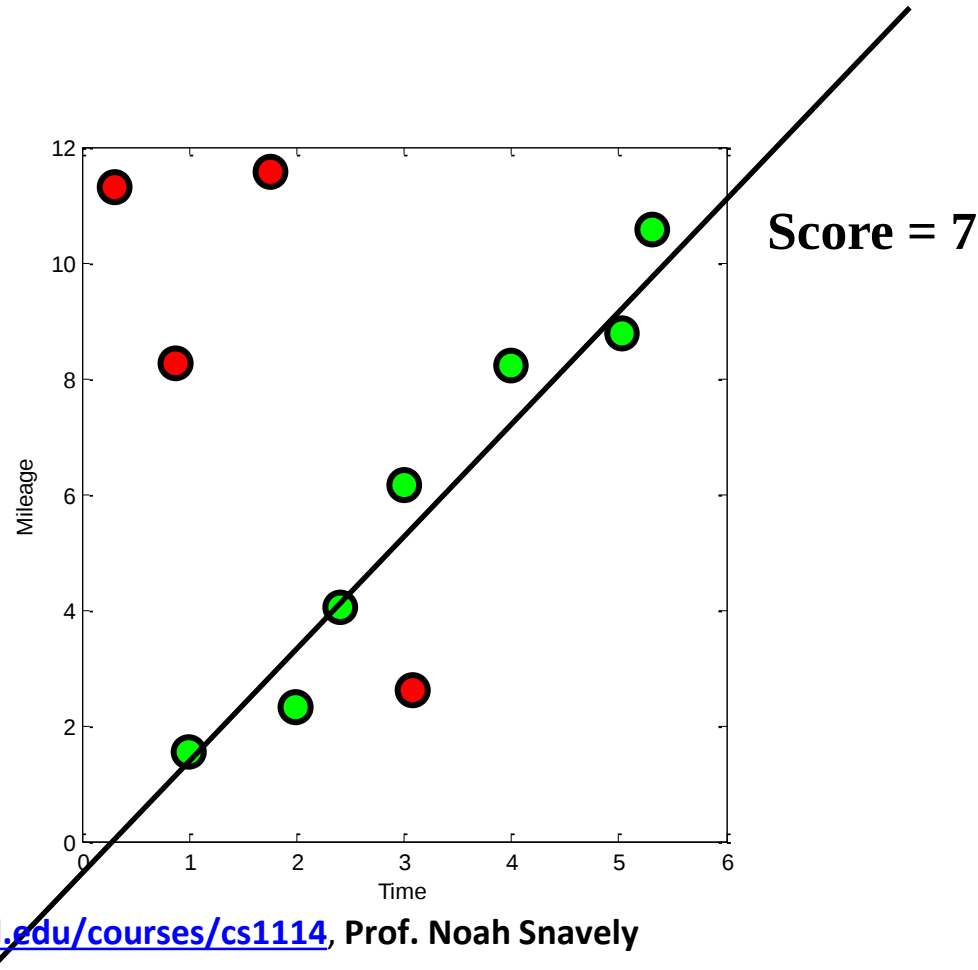
<http://www.cs.cornell.edu/courses/cs1114>, Prof. Noah Snaveley

Testing goodness

- If the distance is small, we call this point an *inlier* to the line
 - If the distance is large, it's an *outlier* to the line
 - For an inlier point and a good line, this distance will be close to (but not exactly) zero
 - For an outlier point or bad line, this distance will probably be large
-
- Objective function: find the line with the most inliers (or the fewest outliers)

Optimizing for inlier count

- How do we find the best possible line?



<http://www.cs.cornell.edu/courses/cs1114>, Prof. Noah Snaveley

RANSAC for estimating homography

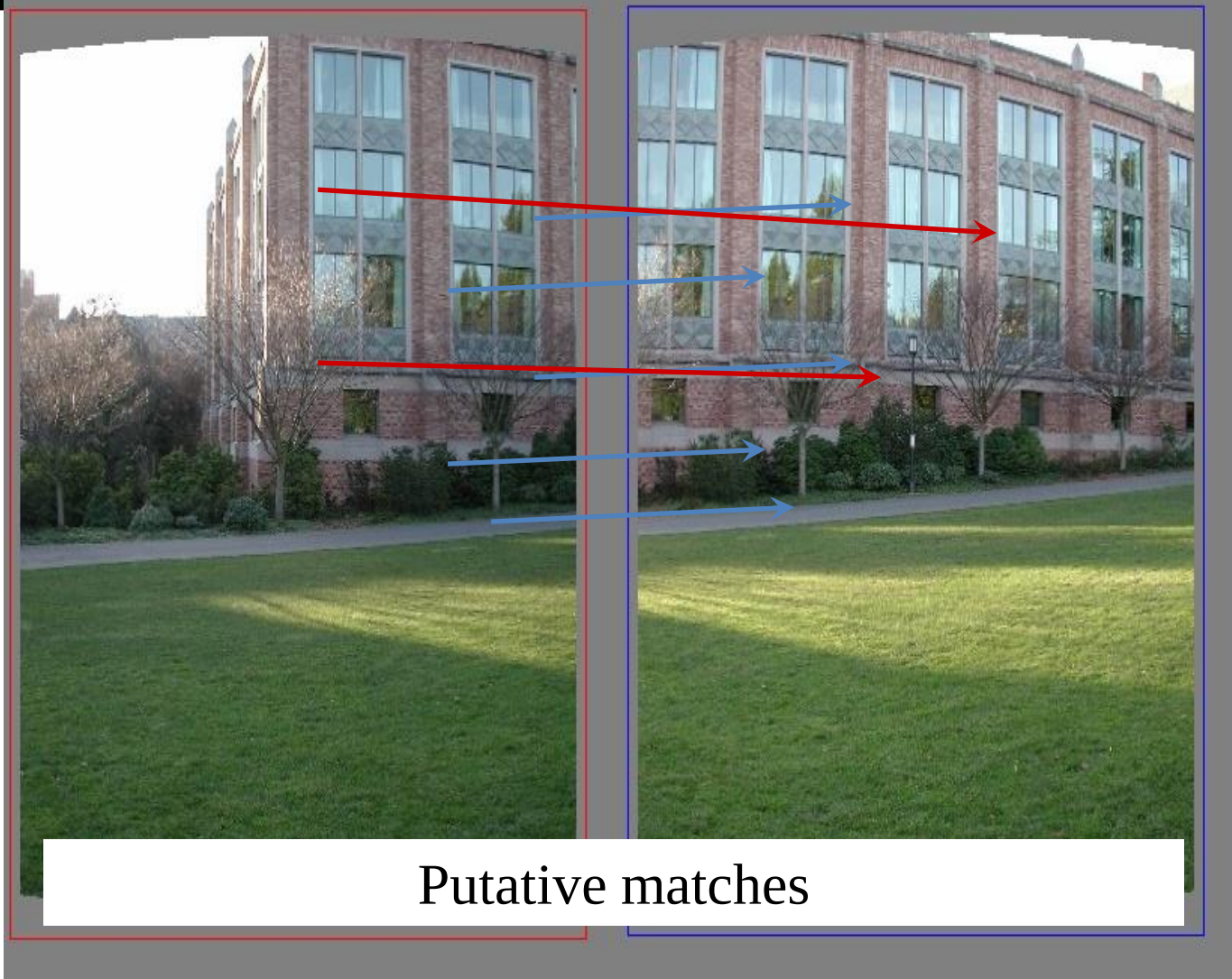
- RANSAC loop:

- 
1. Select four feature pairs (at random)
 2. Compute homography H (exact)
 3. Compute *inliers* where $SSD(p_i', \mathbf{H} p_i) < \varepsilon$
 4. Keep largest set of inliers
 5. Re-compute least-squares H estimate on all of the inliers

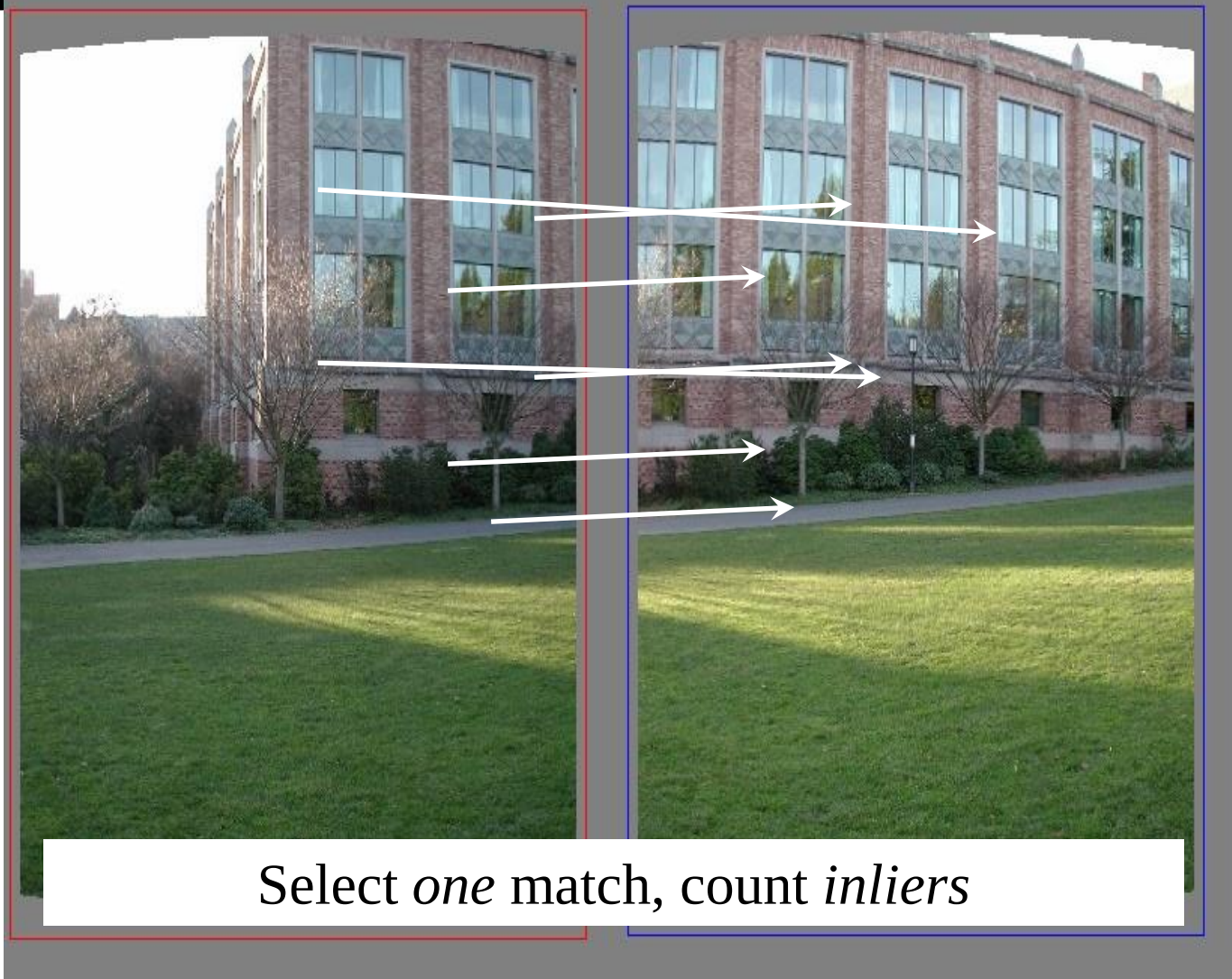
RANSAC

- RANSAC loop:
 1. Randomly select a *seed group* of matches
 2. Compute transformation from seed group
 3. Find *inliers* to this transformation
 4. If the number of inliers is sufficiently large, re-compute least-squares estimate of transformation on all of the inliers
- Keep the transformation with the largest number of inliers

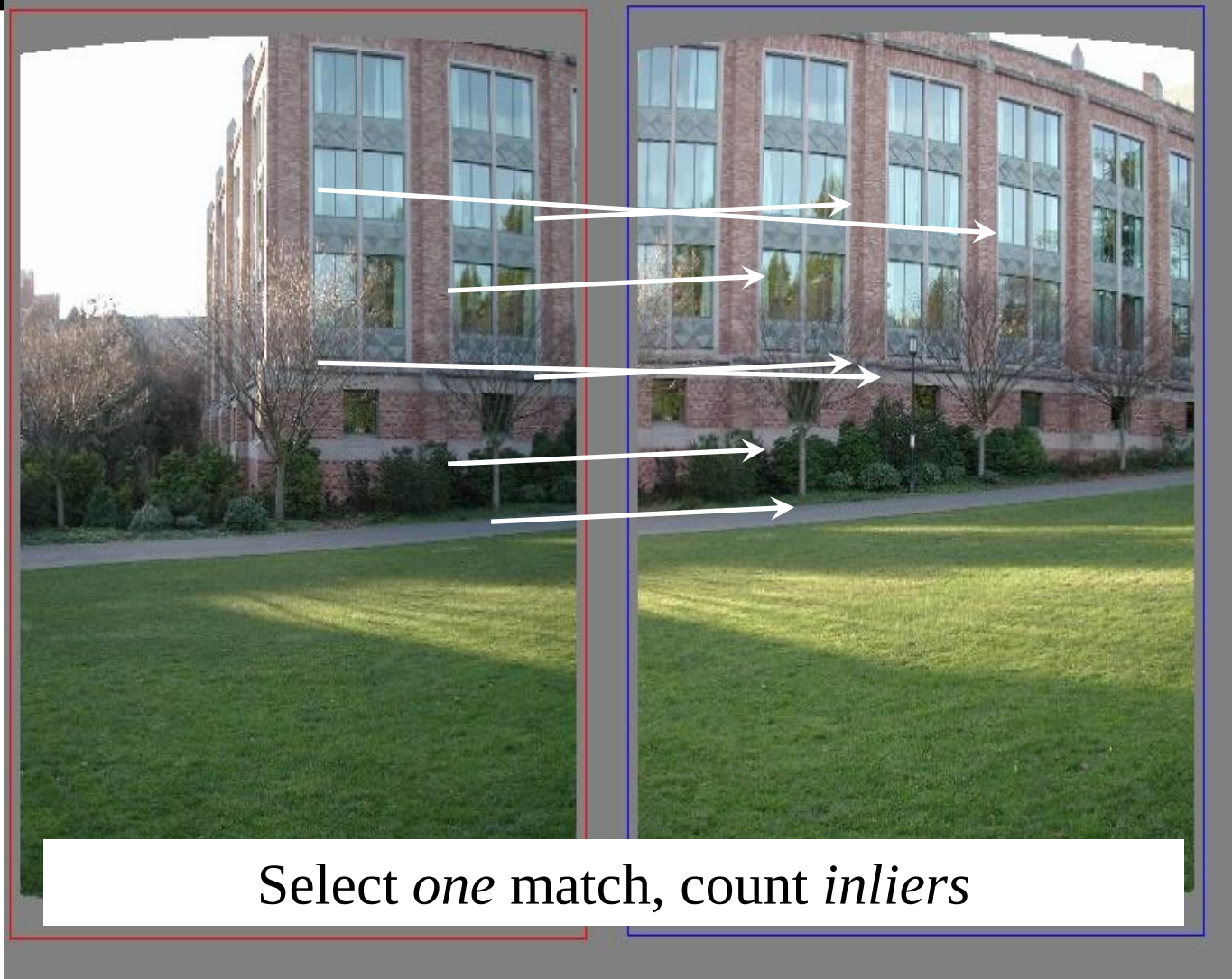
RANSAC example: Translation



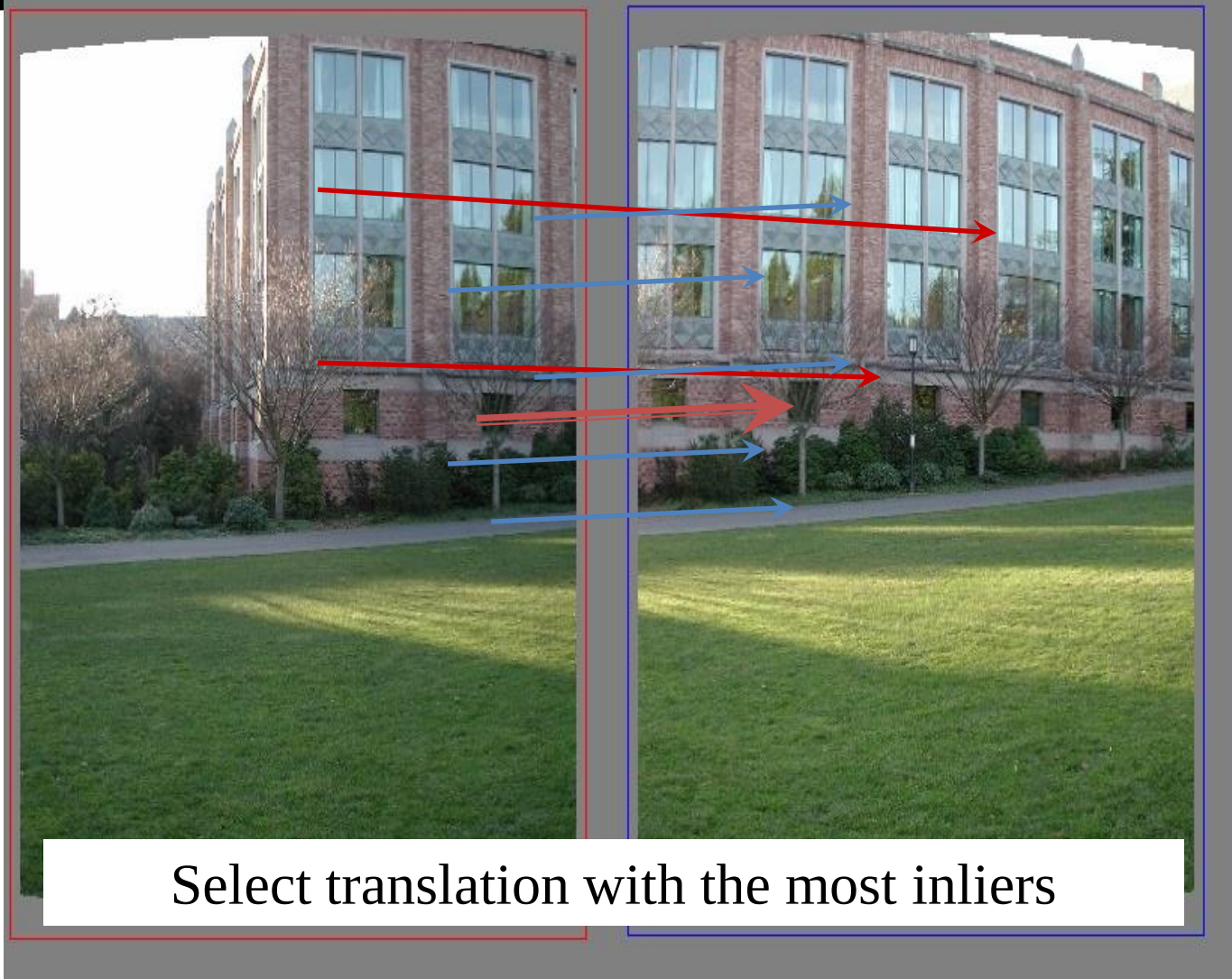
RANSAC example: Translation



RANSAC example: Translation



RANSAC example: Translation



How Many Samples?

On average

- e : probability that a point is an outlier
- s : num of points in sample
- N : num of samples
- p : probability that we get a good sample

- $(1-e)$: prob that one selected point is inlier
- $(1-e)^s$: prob that all selected points are inliers
- $1-(1-e)^s$: prob that not all selected points are inliers (at least 1 outlier)
- $(1-(1-e)^s)^N$: prob that all N selections of s -samples contain at least 1 outlier
- $1 - (1-(1-e)^s)^N$: prob that at least 1 out of N selections of s -samples contain no outlier (it is good).

$$N = \frac{\log(1 - p)}{\log(1 - (1 - e)^s)}$$

proportion of outliers e							
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

How Many Samples

I / N [%]

Size of the sample m

	15%	20%	30%	40%	50%	70%
2	132	73	32	17	10	4
4	5916	1871	368	116	46	11
7	$1.75 \cdot 10^6$	$2.34 \cdot 10^5$	$1.37 \cdot 10^4$	1827	382	35
8	$1.17 \cdot 10^7$	$1.17 \cdot 10^6$	$4.57 \cdot 10^4$	4570	765	50
12	$2.31 \cdot 10^{10}$	$7.31 \cdot 10^8$	$5.64 \cdot 10^6$	$1.79 \cdot 10^5$	$1.23 \cdot 10^4$	215
18	$2.08 \cdot 10^{15}$	$1.14 \cdot 10^{13}$	$7.73 \cdot 10^9$	$4.36 \cdot 10^7$	$7.85 \cdot 10^5$	1838
30	∞	∞	$1.35 \cdot 10^{16}$	$2.60 \cdot 10^{12}$	$3.22 \cdot 10^9$	$1.33 \cdot 10^5$
40	∞	∞	∞	$2.70 \cdot 10^{16}$	$3.29 \cdot 10^{12}$	$4.71 \cdot 10^6$

Τμηματοποίηση γνωστών γεωμετρικών σχημάτων σε εικόνες: Μετασχηματισμός Hough (HT)

- Εστω ότι αναζητούμε σχήμα με γνωστή εξίσωση, πχ ευθεία με εξίσωση:
$$y = mx + c$$
- Ο HT ανιχνεύει γραμμές με τον ακόλουθο τρόπο:
 - Κατασκευάζεται ένας πίνακας (Accumulator) με διάσταση ίση με τον αριθμό παραμέτρων της εξίσωσης του υπό αναζήτηση σχήματος (πχ 2D πίνακας)
 - Ανιχνεύονται τα pixel ακμών $\mathbf{p}$ της εικόνας για κάθε ένα από τα οποία:
 - Κατασκευάζεται το αντίστοιχο σχήμα στον παραμετρικό χώρο (για ευθεία, κατασκευάζεται η ευθεία στον παραμετρικό χώρο m, c που ικανοποιεί τη σχέση της ευθείας) και κάθε pixel του Accumulator από όπου περνά η ευθεία αυξάνει την τιμή του κατά 1
 - Η θέση του pixel με τη μέγιστη τιμή καθορίζει τις παραμέτρους της εξίσωσης του σχήματος που αναζητείται

Μετασχηματισμός Hough (HT): η περίπτωση της ευθείας

- Εστω εικόνα / $N \times M$ στην οποία αναζητάμε ευθείες με εξίσωση

$$y = mx + c$$

- Επειδή ο Accumulator αποτελεί πίνακα με όλες τις δυνατές τιμές των παραμέτρων της ευθείας m, c , καθίσταται απαραίτητη μία αναπαράσταση της ευθείας με παραμέτρους που έχουν φραγμένες τιμές (πχ η παράμετρος m δεν είναι φραγμένη, γιατί αν η κλίση της ευθείας $\rightarrow \pi/2$, τότε η m απειρίζεται).

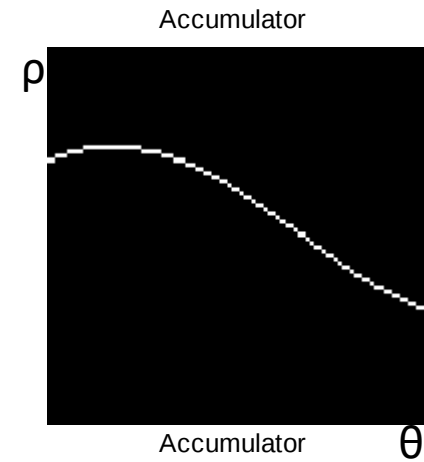
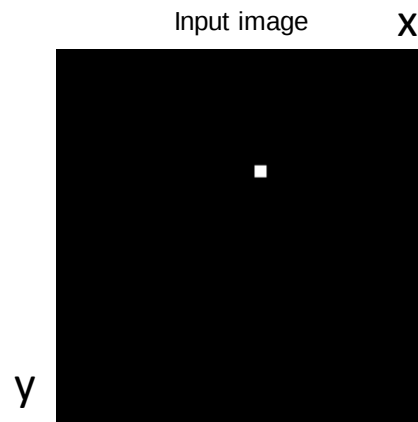
- Εναλλακτική αναπαράσταση ευθείας σε πολικές συντεταγμένες:

$$\rho = x \cos \theta + y \sin \theta$$

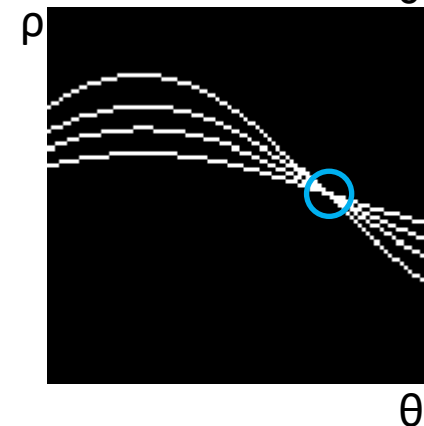
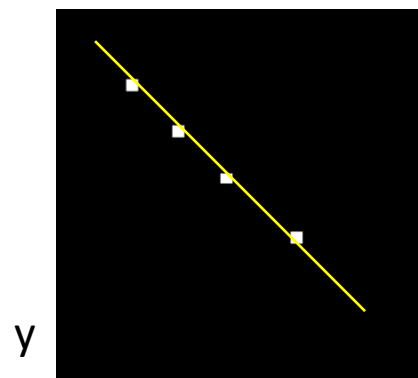
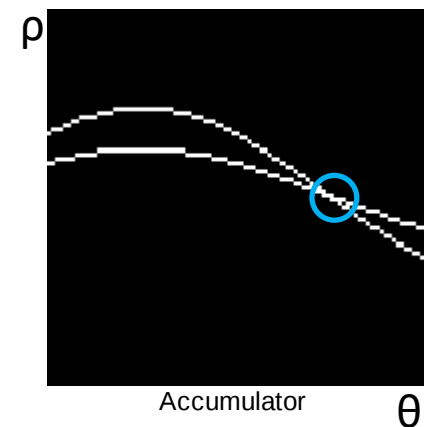
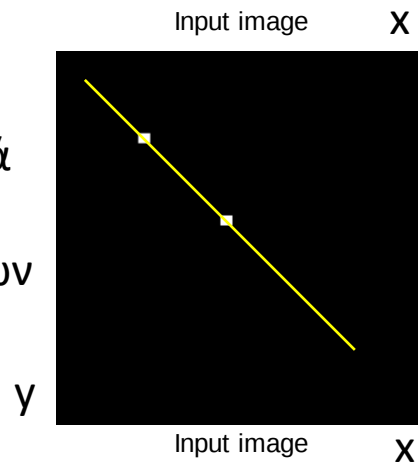
- Ορίζουμε τον αριθμό των τιμών της γωνίας θ από $-\pi$ ως π . Αυτός αποτελεί τις στήλες του Accumulator. Οι γραμμές του πίνακα Accumulator ορίζονται από $-\delta$ ως δ , όπου δ η μέγιστη δυνατή τιμή της παραμέτρου ρ : $\delta = \text{ceil}\left(\sqrt{N^2 + M^2}\right)$

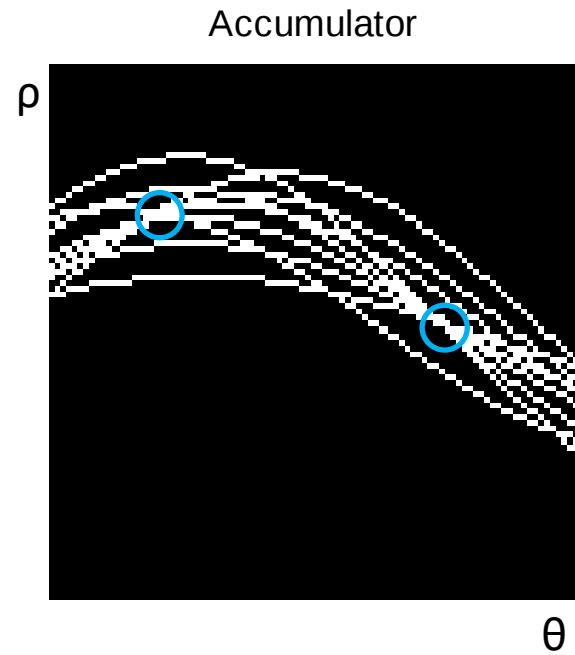
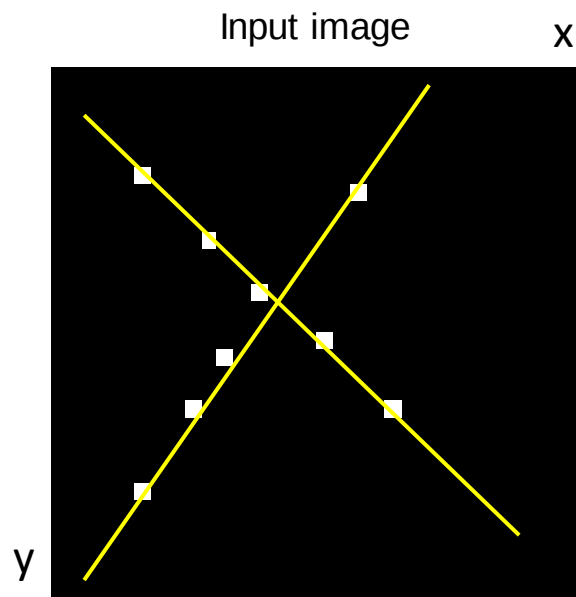
- για κάθε pixel (x, y) της I : $I(x, y) > 0$
 - για κάθε θ
 - Υπολόγισε το $\rho(\theta) = x \cos(\theta) + y \sin(\theta)$
 - $A(\rho, \theta) = A(\rho, \theta) + 1$
 - end
- end
- Κατά συνέπεια: ένα σημείο ακμής στην I παράγει μία καμπύλη στον πίνακα A .
- Αν υπάρχουν k συνευθειακά σημεία, τότε παράγονται k καμπύλες οι οποίες τέμνονται σε ένα σημείο του A . Το σημείο αυτό θα έχει τη μέγιστη τιμή από όλα τα pixel του A και οι συντεταγμένες του καθορίζουν τις παραμέτρους της ευθείας που διέρχεται από τα σημεία της I .

Αρχική εικόνα με 1 σημείο >0
και η αναπαράσταση του στον
Accumulator



Αρχική εικόνα με 2 και 4 συνευθειακά
σημεία >0 και η αναπαράσταση
τους στον Accumulator. Η τομή των
αντίστοιχων ημιτονοειδών
καμπύλων ορίζει το pixel με
μέγιστη τιμή=2 και 4<αντίστοιχα.
Οι συντεταγμένες του ορίζουν τις
παραμέτρους της ευθείας που
περνά από τα σημεία.





Iterative Closest Point (ICP)

Goal: estimate transform between two dense sets of points

1. **Initialize** transformation (e.g., compute difference in means and scale)
2. **Assign** each point in {Set 1} to its nearest neighbor in {Set 2}
3. **Estimate** transformation parameters
 - e.g., least squares or robust least squares
4. **Transform** the points in {Set 1} using estimated parameters
5. **Repeat** steps 2-4 until change is very small

<https://www.youtube.com/watch?v=m64E47uvPYc>

https://www.youtube.com/watch?v=uzOCS_gdZuM