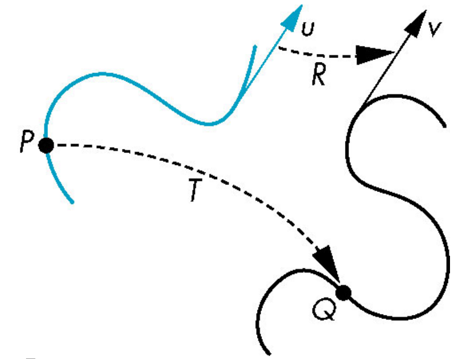
Μετασχηματισμοί



- Μετασχηματισμός: απεικόνιση ενός σημείου ή διανύσματος σε άλλο σημείο ή διάνυσμα Q=T(P), v=R(u)
- Ομογενείς συντεταγμένες: ενιαίος ορισμός
- **q**=f(**p**)
- Γενική περίπτωση: υπολογισμός για κάθε σημείο ευθείας/καμπύλης χωριστά.

Συναφείς μετασχηματισμοί

- Συναφείς μετασχηματισμοί:
- Απεικονίζουν σημεία σε σημεία και διανύσματα σε διανύσματα.
- $T(\alpha u+\beta v)=\alpha T(u)+\beta T(v)$
- $T(P+\alpha u)=T(P)+\alpha T(u)$
  – Διατηρούν τις ευθείες, τους λόγους αποστάσεων και την παραλληλία
  – Δεν διατηρούν πάντα αποστάσεις & γωνίες

Συναφείς μετασχηματισμοί

- Στις ομογενείς συντεταγμένες μπορεί να γραφούν σαν πράξη πινάκων:
- **v**=**Au**, **A**: 4x4 πίνακας

$$\mathbf{A} = \begin{bmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} & \gamma_{14} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} & \gamma_{24} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} & \gamma_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Μετασχηματισμός σημείων / διανυσμάτων: 12 / 9 βαθμοί ελευθερίας (δεν έχει νόημα η μετατόπιση διανύσματος)

Συναφείς μετασχηματισμοί

- Δύο σκοπιές για να δούμε τους μετασχηματισμούς:
- Αλλαγή πλαισίου συντεταγμένων
- Μετακίνηση των σημείων & διανυσμάτων στο ίδιο πλαίσιο

Συναφείς μετασχηματισμοί

- $\mathbf{p}(\alpha)=\alpha\mathbf{p}_0+(1-\alpha)\ \mathbf{p}_1$
- $\mathbf{Ap}(\alpha)=\alpha\mathbf{Ap}_0+(1-\alpha)\mathbf{A}\ \mathbf{p}_1$
- Υπολογίζω μόνο τους μετασχηματισμούς των $\mathbf{p}_0$, $\mathbf{p}_1$, όχι όλα τα σημεία της ευθείας

Μετασχηματισμοί γραφικών

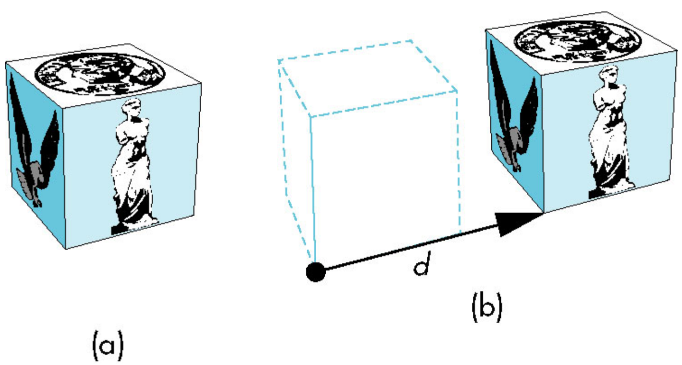
- Πρακτική σημασία συναφών μετασχηματισμών
- Μετασχηματισμοί που χρειαζόμαστε: μετατόπιση, περιστροφή, κλιμάκωση, στρέβλωση
  - Όλοι είναι συναφείς μετασχηματισμοί
- Με συνδυασμό κλιμακώσεων, περιστροφών & μετατοπίσεων μπορώ να παράξω όλους τους συναφείς μετασχηματισμούς

Μετασχηματισμοί γραφικών

- Εκφράσεις μετασχηματισμών
  - ανεξάρτητες από αναπαράσταση σε πλαίσιο
  - Με τη μορφή πινάκων για χρήση με αναπαραστάσεις σε πλαίσιο & ομ. συντεταγμένες

Μετατόπιση

- Μετακίνηση σημείου κατά συγκεκριμένη απόσταση προς συγκεκριμένη διεύθυνση
- Απαιτεί καθορισμό του διανύσματος μετατόπισης
- P’=P+d
- Τρεις βαθμοί ελευθερίας

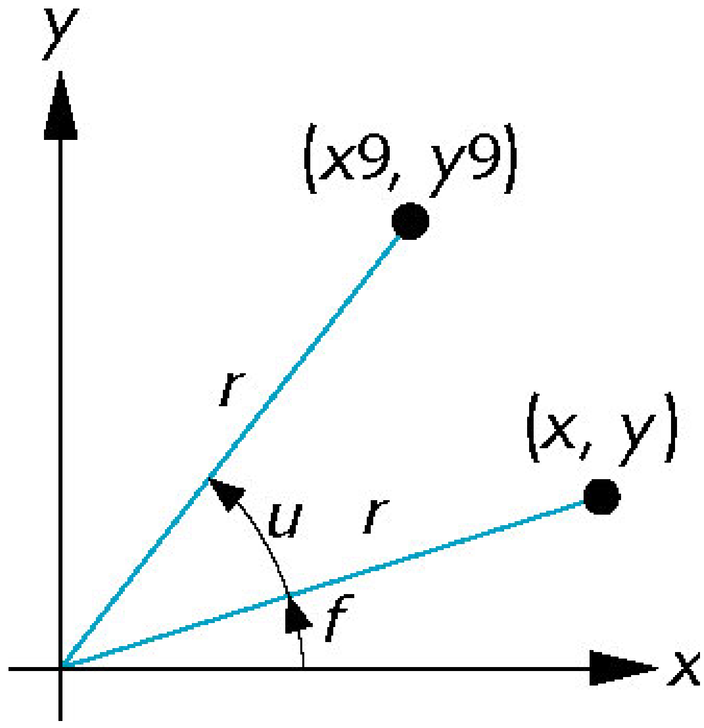


Περιστροφή

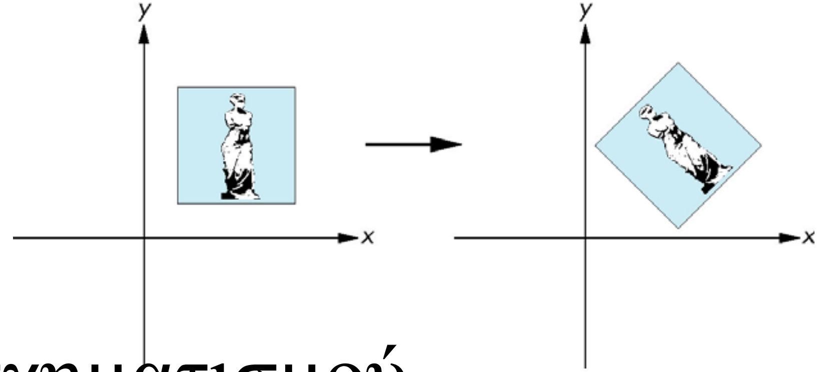
- Απλή περίπτωση: περιστροφή γύρω από την αρχή σε 2 διαστάσεις
- x=ρcos(φ), y=ρsin(φ) (αρχικό)
- x’=ρcos(θ+φ), y’=ρsin(θ+φ) (περιστραμμένο)
- x’=xcos(θ)-ysin(θ), y’=xsin(θ)+ycos(θ)

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Περιστροφή

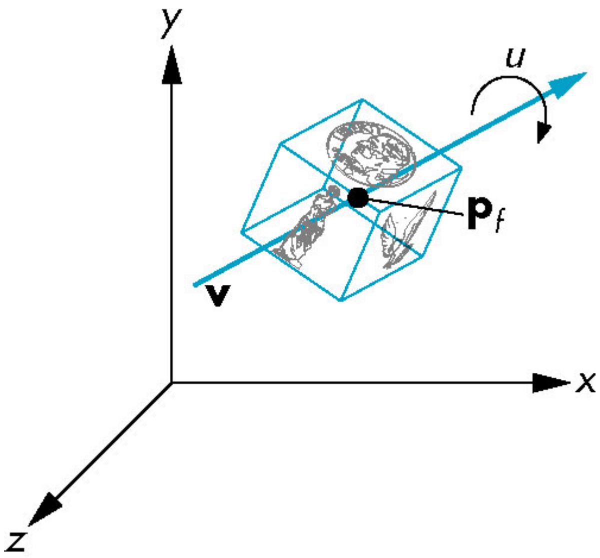


Περιστροφή



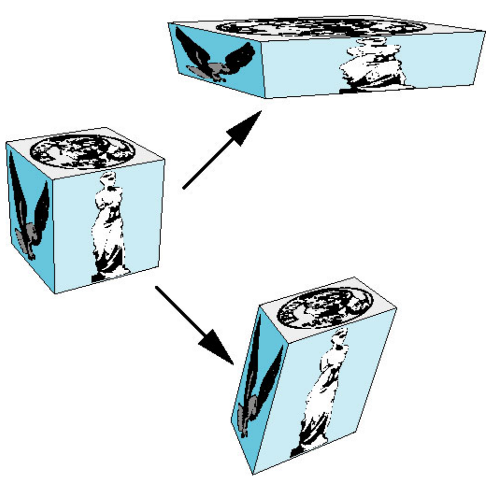
- Σταθερό σημείο μετασχηματισμού
- Υποπερίπτωση τρισδιάστατης περιστροφής
  - Περιστροφή γύρω από τον άξονα z (z αναλλοίωτο)
  - Θετική φορά περιστροφής: ανθωρολογιακή όταν κοιτάζουμε την αρχή από τον θετικό ημιάξονα z
- Γενική μορφή περιστροφής σε 3 διαστάσεις
- Ορισμός σταθερού σημείου, διάνυσμα περιστροφής (μόνο διεύθυνση-φορά, 2 παράμετροι), γωνία περιστροφής (1 παράμετρος)

Περιστροφή



Μετασχηματισμοί στερεού σώματος (rigid body)

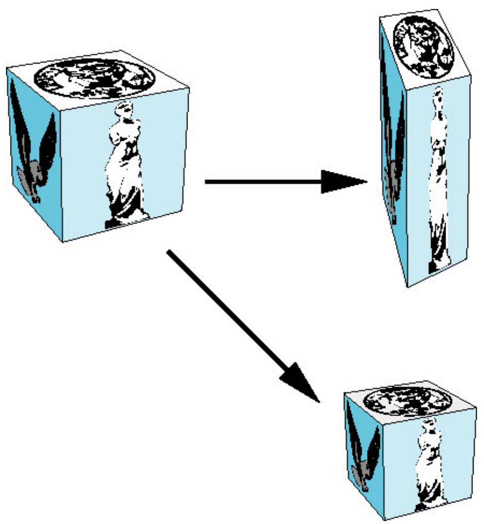
- Δεν αλλοιώνουν το σχήμα ενός αντικειμένου
  - Διατηρούν αποστάσεις & γωνίες
  - Μετατόπιση & περιστροφή
- Υποσύνολο των συναφών μετασχηματισμών

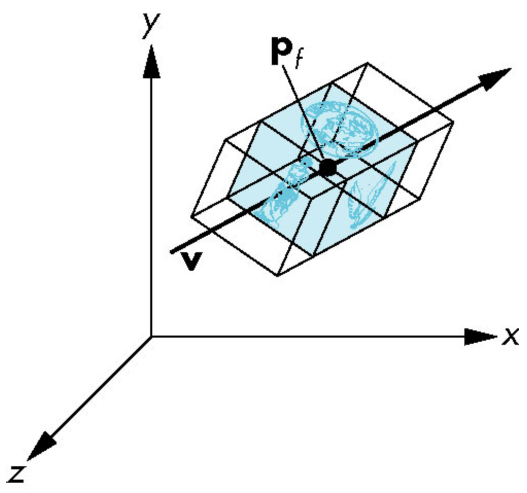


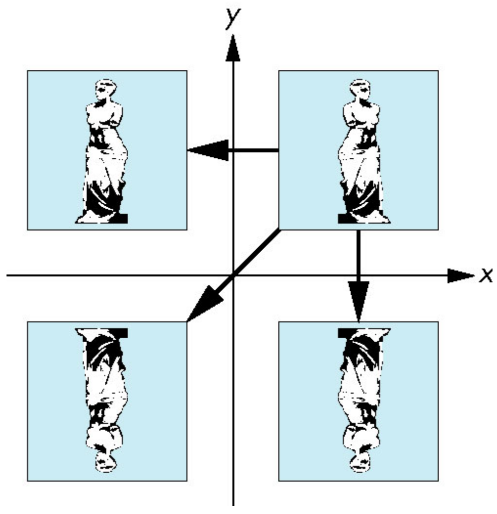
Κλιμάκωση (scaling)

- Συναφής, μη στερεού σώματος μετασχηματισμός
- Σταθερό σημείο, διεύθυνση κλιμάκωσης (συνήθως κάποιος από τους άξονες)
- Μεγέθυνση (α>1), σμίκρυνση (0<α<1), ανάκλαση (α<0)
- Για κλιμάκωση σε πολλές διευθύνσεις: ομοιόμορφη, μη ομοιόμορφη κλιμάκωση

Κλιμάκωση (scaling)







Μετατόπιση

- **p'=p+d**

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} + \begin{bmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \\ 0 \end{bmatrix} \qquad \mathbf{T}(\alpha_x, \alpha_y, \alpha_z) = \begin{bmatrix} 1 & 0 & 0 & \alpha_x \\ 0 & 1 & 0 & \alpha_y \\ 0 & 0 & 1 & \alpha_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- **p'=Tp**
- Χρησιμότητα ομογενών συντεταγμένων

$$\mathbf{T}^{-1}(\alpha_x, \alpha_y, \alpha_z) = \mathbf{T}(-\alpha_x, -\alpha_y, -\alpha_z)$$

Κλιμάκωση

- Κλιμάκωση με σταθερό σημείο την αρχή και ως προς τους τρεις άξονες x,y,z
- x’=β_xx, y’=β_yy, z’=β_zz,

$$\mathbf{S}(\beta_x, \beta_y, \beta_z) = \begin{bmatrix} \beta_x & 0 & 0 & 0 \\ 0 & \beta_y & 0 & 0 \\ 0 & 0 & \beta_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{S}^{-1}(\beta_x, \beta_y, \beta_z) = \mathbf{S}(1/\beta_x, 1/\beta_y, 1/\beta_z)$$

- $\beta_x=\beta_y=\beta_z$ ομοιόμορφη, $\beta_y=\beta_z=1$ μόνο στον x

Περιστροφή

- Τρισδιάστατη περιστροφή ως προς σταθερό σημείο την αρχή, ανεξάρτητα ως προς τους τρεις άξονες

Περιστροφή ως προς τον z

- $x' = x\cos(\theta) - y\sin(\theta)$, $y' = x\sin(\theta) + y\cos(\theta)$, $z' = z$

Περιστροφή

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

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

Περιστροφή

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

$$\mathbf{R}^{-1}(\theta) = \mathbf{R}^{\mathrm{T}}(\theta) = \mathbf{R}(-\theta)$$

- **R:** ορθογώνιος

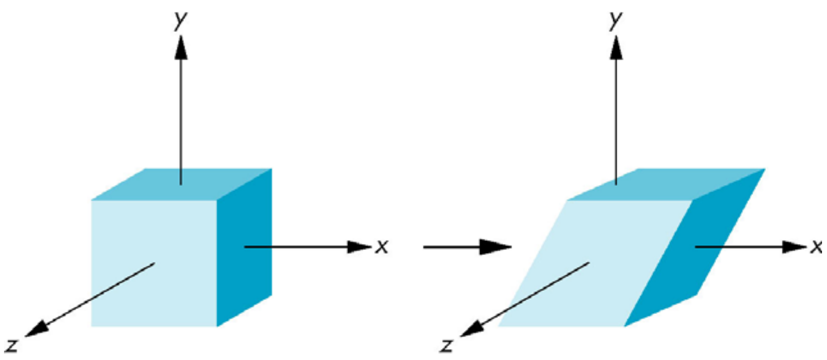
Στρέβλωση (Shear)

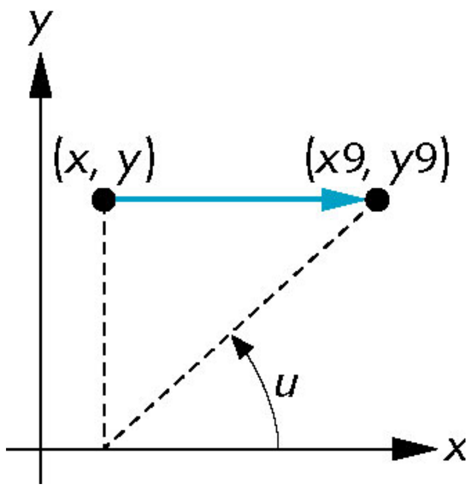
- Προς συγκεκριμένο άξονα, οι συντεταγμένες στους άλλους άξονες δεν αλλάζουν
- Χαρακτηρίζεται από μια γωνία στρέβλωσης
- x’=x+ycot(θ), y’=y, z’=z

$$\mathbf{H}_{x_y}(\theta) = \begin{bmatrix} 1 & \cot(\theta) & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{H}_{x_y}^{-1}(\theta) = \mathbf{H}_{x_y}(-\theta)$$

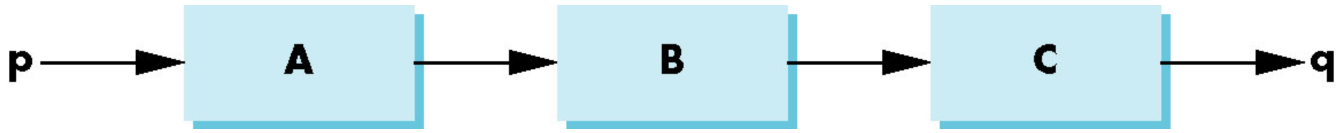
Στρέβλωση (Shear)





Σύνθεση μετασχηματισμών

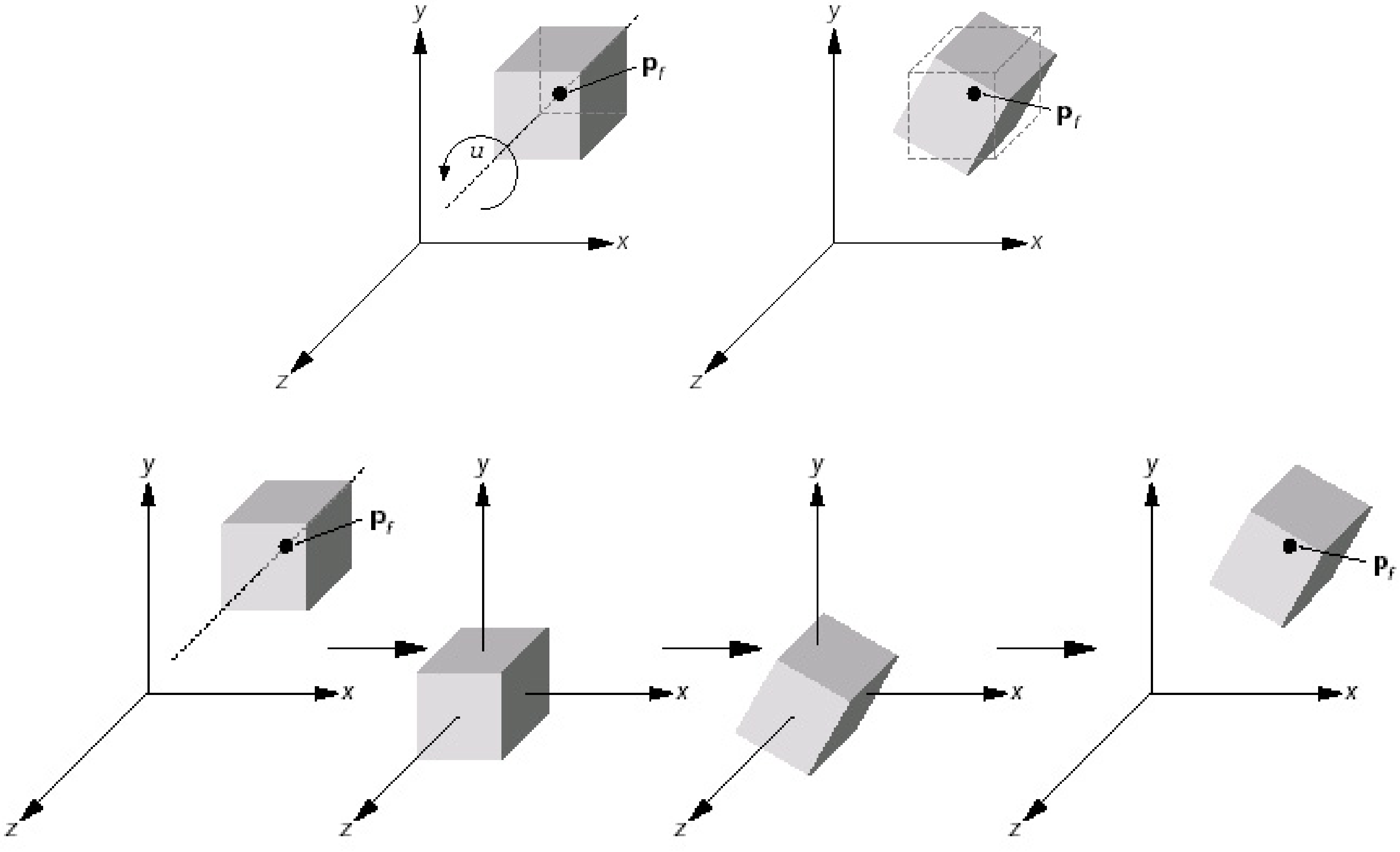
- Έχοντας εξοπλιστεί με τους βασικούς μετασχηματισμούς προχωράω να κατασκευάσω πιο σύνθετους.
- Δημιουργία μετασχηματισμών από σύνθεση επιμέρους βασικών μετασχηματισμών
- Πολλαπλασιασμός των αντίστοιχων πινάκων
- **q=CBAp=C(B(Ap)=(CBA)p**
- Η σειρά παίζει ρόλο!



Σύνθεση μετασχηματισμών

- Δεν μπορώ να αλλάξω τη σειρά των μετασχηματισμών αλλά μπορώ να αλλάξω τη σειρά των υπολογισμών
- Παράδειγμα: περιστροφή κύβου με σταθερό σημείο το κέντρο του **p** και γύρω από τον z.
  - Γνωρίζω πως να κάνω περιστροφή με σταθερό σημείο την αρχή και γύρω από τον z.
  - Μετατοπίζω τον κύβο στην αρχή **T**(-**p**)
  - Περιστρέφω γύρω από z $\mathbf{R}_z(\theta)$
  - Ξανατοποθετώ τον κύβο εκεί που ήταν **T**(**p**)

Σύνθεση μετασχηματισμών



Σύνθεση μετασχηματισμών

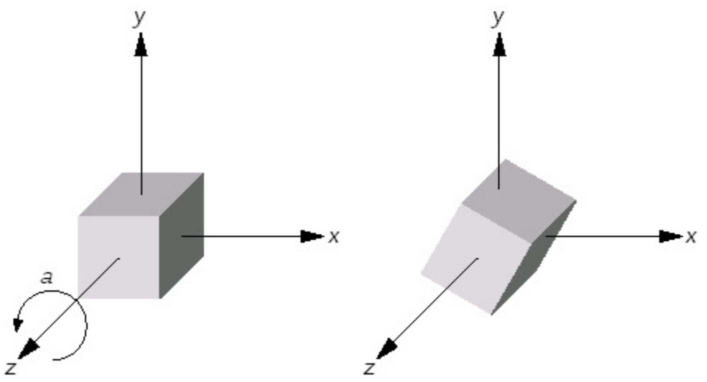
– $\mathbf{M}=\mathbf{T}(\mathbf{p})\ \mathbf{R}_z(\theta)\ \mathbf{T}(\mathbf{-p})$

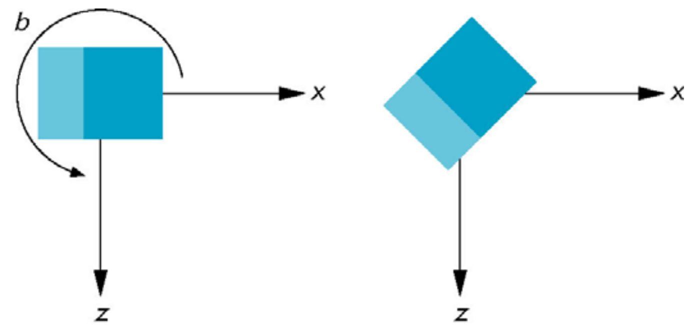
$$\mathbf{M}=\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & x-x\cos(\theta)+y\sin(\theta) \\ \sin(\theta) & \cos(\theta) & 0 & y-x\sin(\theta)-y\cos(\theta) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Περιστροφή περί την αρχή

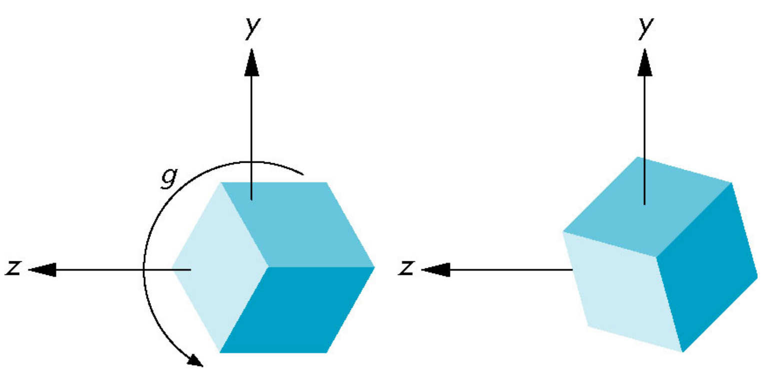
- Οποιαδήποτε περιστροφή γύρω από την αρχή μπορεί να συντεθεί ως 3 ανεξάρτητες περιστροφές ως προς τους 3 άξονες
- $\mathbf{M} = \mathbf{R}_x(\alpha)\, \mathbf{R}_y(\beta)\, \mathbf{R}_z(\gamma)$
- Ο προσδιορισμός των α,β,γ δεν είναι εύκολος

Περιστροφή περί την αρχή





(a) (b)

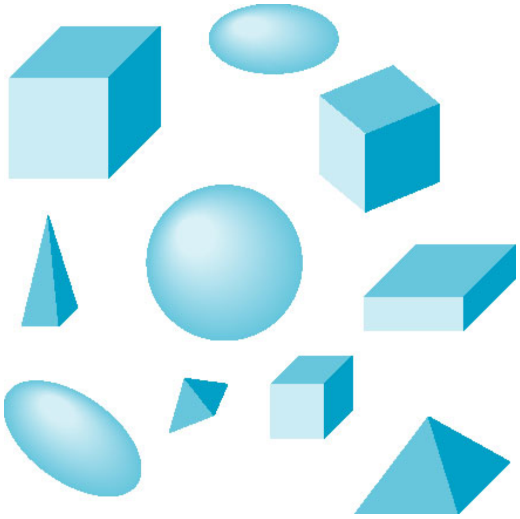


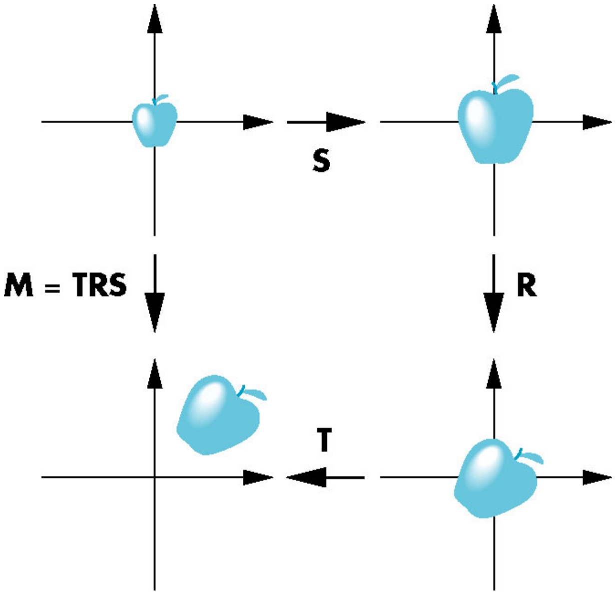
(a) (b)

Μετασχηματισμός στιγμιοτύπου (instance transform)

- Τοποθέτηση αντικειμένων εκεί που θέλουμε
- Ορισμός των αντικειμένων σε βολική θέση (βάση αντικειμένων) και συνδυασμός μετασχηματισμών για να τα τοποθετήσουμε στην τελική τους θέση (στιγμιότυπο)
- **M=TRS**
- Χρήση display lists

Μετασχηματισμός στιγμιοτύπου (instance transform)

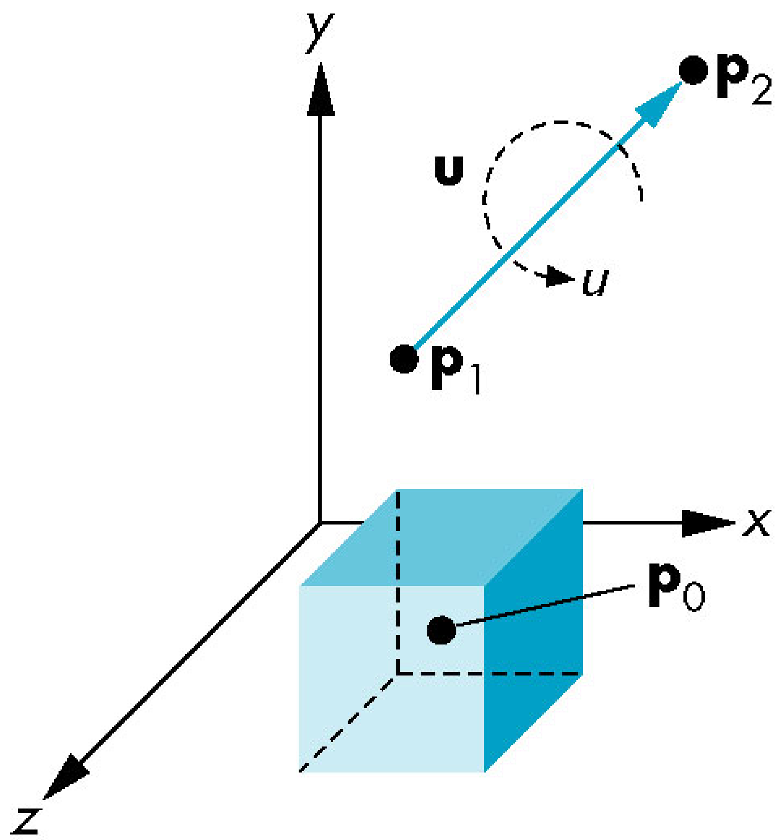




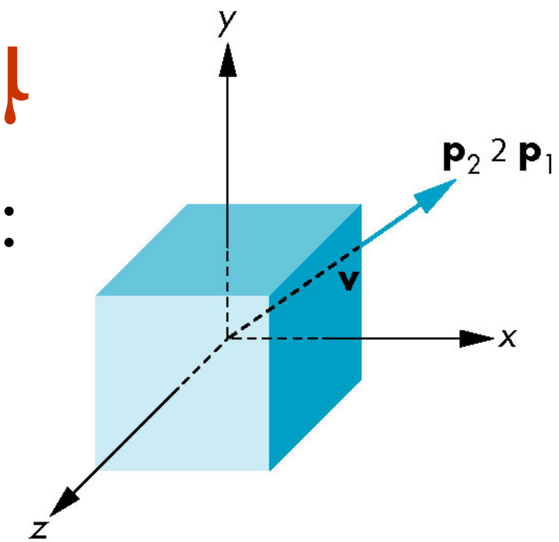
Περιστροφή ως προς τυχαίο διάνυσμα και σημείο

- Σημείο περιστροφής $\mathbf{p}_0$, διάνυσμα (άξονας) περιστροφής **u**, γωνία περιστροφής θ
- Ορισμός του διανύσματος περιστροφής **u** από δύο σημεία: $\mathbf{u}=\mathbf{p}_2-\mathbf{p}_1$
- Η διάταξη των σημείων ορίζει την θετική περιστροφή
- Ενδιαφέρει μόνο η διεύθυνση του **u**, όχι το μέτρο

Περιστροφή ως προς τυχαίο διάνυσμα και σημείο



Περιστροφή ως προς τυχαίο διάνυσμα και σημ



- Κάνω το διάνυσμα μοναδιαίο:

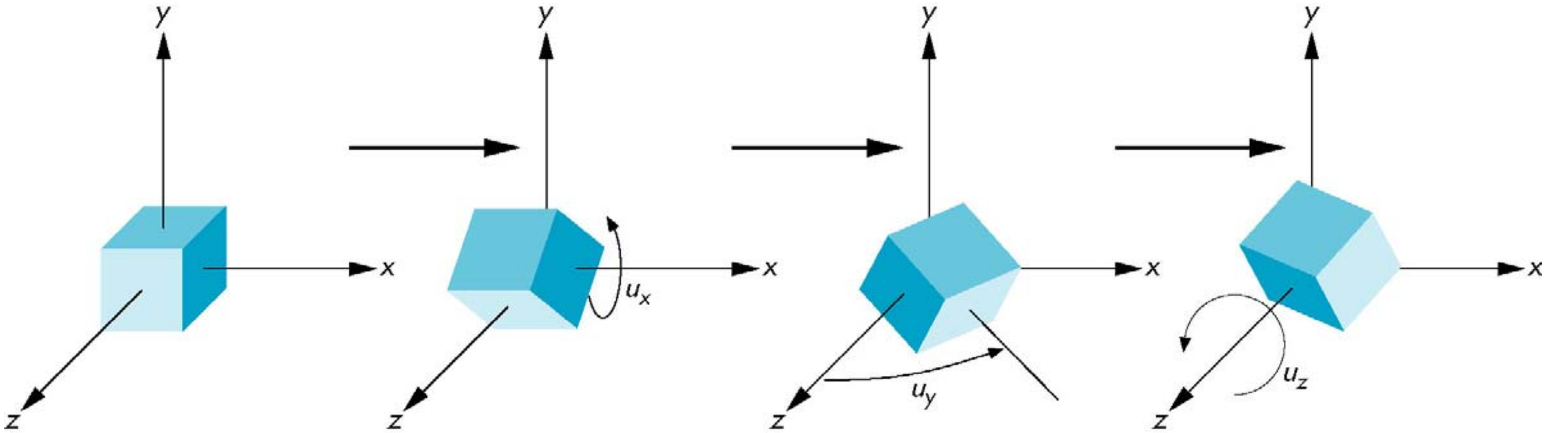
$$\mathbf{v} = \frac{\mathbf{u}}{|\,\mathbf{u}\,|} = \begin{bmatrix} \alpha_x \\ \alpha_y \\ \alpha_z \end{bmatrix}$$

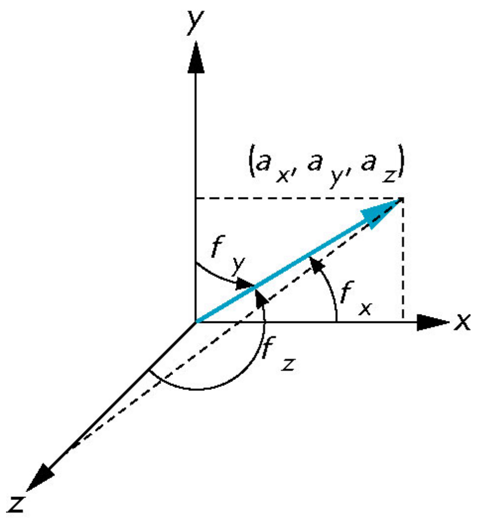
- Αρχίζω με μετατόπιση του $\mathbf{p}_0$ στην αρχή $\mathbf{T}(-\mathbf{p}_0)$, τελειώνω με επανατοποθέτηση στο $\mathbf{p}_0$ $\mathbf{T}(\mathbf{p}_0)$

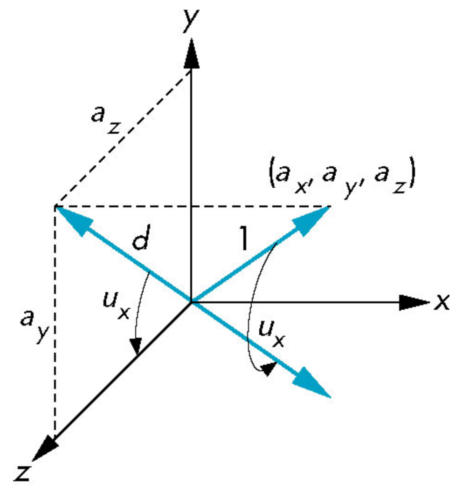
Περιστροφή ως προς τυχαίο διάνυσμα και σημείο

- Γνωρίζω πως να κάνω περιστροφή γύρω από έναν άξονα και την αρχή των αξόνων.
- Στρατηγική: δύο περιστροφές ως προς x & y ώστε να ευθυγραμμίσω το **v** με τον άξονα z, περιστροφή θ και αντιστροφή των δύο περιστροφών.
- $\mathbf{R} = \mathbf{R}_x(-\theta_x)\ \mathbf{R}_y(-\theta_y)\ \mathbf{R}_z(\theta)\ \mathbf{R}_y(\theta_y)\ \mathbf{R}_x(\theta_x)$
- Πώς βρίσκω τα θ_x , θ_y ?

Περιστροφή ως προς τυχαίο διάνυσμα και σημείο







Περιστροφή ως προς τυχαίο διάνυσμα και σημείο

- Γωνίες με τους άξονες: γωνίες κατεύθυνσης φ_x φ_y φ_z
- Συνημίτονα κατεύθυνσης:
- $\cos\varphi_x = \alpha_x$, $\cos\varphi_y = \alpha_y$, $\cos\varphi_z = \alpha_z$
- $\cos^2 \varphi_x + \cos^2 \varphi_y + \cos^2 \varphi_z = 1$
- Η περιστροφή ως προς τον x θα πρέπει να είναι τέτοια που να φέρει το **v** στο επίπεδο y=0 (xz)
- Ισοδύναμα η προβολή του **v** στο επίπεδο x=0 (yz) θα πρέπει να πέσει στον άξονα z
- Μήκος της προβολής $d = \sqrt{\alpha_y^2 + \alpha_z^2}$

Περιστροφή ως προς τυχαίο διάνυσμα και σημείο

- Αρκεί να προσδιορίσω τα $\cos\theta_x$, $\sin\theta_x$, όχι το θ_x: $\cos\theta_x = \alpha_z/d$, $\sin\theta_x = \alpha_y/d$

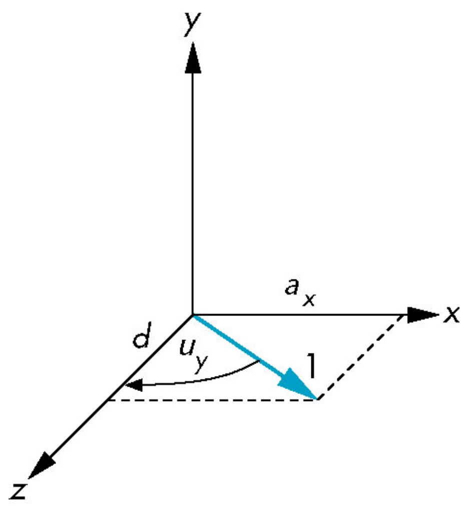
$$\mathbf{R}_x(\theta_x) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \alpha_z/d & -\alpha_y/d & 0 \\ 0 & \alpha_y/d & \alpha_z/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Το **v** είναι τώρα πάνω στο επίπεδο y=0

Περιστροφή ως προς τυχαίο διάνυσμα και σημείο

- Η γωνία είναι τώρα προς τη φορά του ωρολογίου (αρνητικό πρόσημο)
- $\cos(-\theta_y) = d$, $\sin(-\theta_y) = -\alpha_x$

$$\mathbf{R}_y(\theta_y) = \begin{bmatrix} d & 0 & -\alpha_x & 0 \\ 0 & 1 & 0 & 0 \\ \alpha_x & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

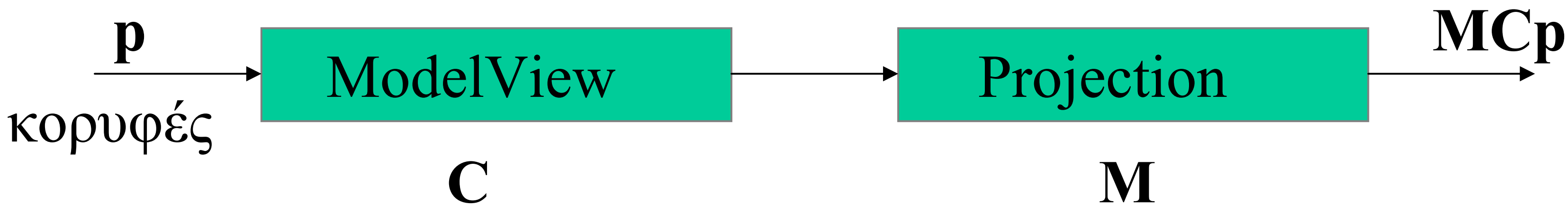


OpenGL & πίνακες μετασχηματισμών

- Τρεις πίνακες μέρος της κατάστασης:
  - Modelview (3-D σε 3-D):
    - Μετασχηματίζει τα σημεία των αντικειμένων (προσέγγιση ενός πλαισίου)
    - Μετακινεί το κοσμικό πλαίσιο ως προς το πλαίσιο της κάμερας (προσέγγιση δύο πλαισίων)
  - Projection (3-D σε 2-D)
    - Προβάλλει τα σημεία στο επίπεδο του φιλμ
  - Texture

OpenGL & πίνακες μετασχηματισμών

- Projection*Modelview: τρέχων πίνακας μετασχηματισμού (CTM): εφαρμόζεται σε όλες τις κορυφές που σχεδιάζω.



- Αν μεταβάλλω τον πίνακα CTM ο τροποποιημένος πίνακας εφαρμόζεται σε όλες τις κορυφές που θα ορισθούν εφεξής
- **p’=MCp**

OpenGL & πίνακες μετασχηματισμών

- Θα ασχοληθούμε μόνο με τον ModelView
- Οι πίνακες αλλάζουν με ίδιο σετ εντολών
- Επιλογή του πίνακα που θα αλλάξω: glMatrixMode
- Αρχικά ο **C** είναι ο μοναδιαίος: τα σημεία μετασχηματίζονται στον εαυτό τους

Είδη εντολών για πίνακες

- Απευθείας ορισμός του πίνακα
  - glLoadIdentity(): αρχικοποίηση σε μοναδιαίο
  - glLoadMatrix(**M**): θέσε ως **C** τον **M**

$$\mathbf{C} \leftarrow \mathbf{M}$$

  - **M** μονοδιάστατος, κατά στήλες

Είδη εντολών για πίνακες

- Πολλαπλασιασμός του τρέχοντα πίνακα **C** από δεξιά με κάποιον πίνακα **M**
  - glMultMatrix(**M**):

    $$\mathbf{C} \leftarrow \mathbf{CM}$$

  - **M** μονοδιάστατος, κατά στήλες
  - glRotatef, glTranslatef, glScalef: υπολογίζουν τον αντίστοιχο πίνακα μετασχηματισμού και τον πολλαπλασιάζουν από δεξιά με τον C

Είδη εντολών για πίνακες

- glTranslatef(dx, dy, dz)
  - Μετατόπιση κατά διάνυσμα dx, dy, dz (κατά dx, dy, dz στους αντίστοιχους άξονες)

    $\mathbf{C} \leftarrow \mathbf{CT}$
  - Π.χ: glTranslatef(3, 0,0): δημιουργία του:

$$\mathbf{T}(3,0,0) = \begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

  - Πολλαπλασιασμός με τον **C**

Είδη εντολών για πίνακες

- glRotatef(angle, vx, vy, vz)
  - Περιστροφή κατά angle περί την αρχή και περί το διάνυσμα (vx, vy, vz) (διάνυσμα από την αρχή στο σημείο (vx, vy, vz)

    $$\mathbf{C} \leftarrow \mathbf{CR}$$

- glScalef(sx,sy,sz): κλιμάκωση κατά sx,sy,sz στους αντίστοιχους άξονες

  $$\mathbf{C} \leftarrow \mathbf{CS}$$

Η θεώρηση του ενός πλαισίου

- Η κάμερα στην αρχή του πλαισίου, «κοιτάζει» στον αρνητικό z, «πάνω» ο θετικός y.
- Ο ModelView μετακινεί τις κορυφές στο ενιαίο σύστημα.

- Περιστροφή κατά 45^0 με κέντρο (4,5,6) και γύρω από το διάνυσμα (1,2,3)

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(4, 5, 6);
glRotatef(45, 1, 2, 3);
glTranslatef(-4, -5, -6);
```

Σειρά εφαρμογής μετασχηματισμών

- Οι μετασχηματισμοί εφαρμόζονται στα σημεία με σειρά αντίθετη από αυτή με την οποία ορίζονται στο πρόγραμμα!

$$\mathbf{C} \leftarrow \mathbf{I}$$

$$\mathbf{C} \leftarrow \mathbf{CT}(4,5,6)$$

$$\mathbf{C} \leftarrow \mathbf{CR}(45,1,2,3)$$

$$\mathbf{C} \leftarrow \mathbf{CT}(-4,-5,-6)$$

$$\mathbf{C} = \mathbf{T}(4,5,6)\mathbf{R}(45,1,2,3)\mathbf{T}(-4,-5,-6)$$

$$\mathbf{p}' = \mathbf{Cp}$$

Σημεία προσοχής

- Ο τρέχων πίνακας μετασχηματισμού εφαρμόζεται σε όλες τις κορυφές που ορίζω από εκεί και κάτω
- Οι μετασχηματισμοί δρουν συσσωρευτικά
- Αν θέλω κάποιες κορυφές να μην μετασχηματίζονται: glLoadIdentity()
- Αν θέλω να «απομονώσω» μετασχηματισμούς (να εφαρμόζονται σε συγκεκριμένο σύνολο σημείων): glPushMatrix(), glPopMatrix()

Παράδειγμα

- Τοποθέτηση 4 κύβων σε τέσσερις κορυφές ενός τετραγώνου, μετατόπιση όλου του « συστήματος» κατά έναν άξονα
  - Εκκίνηση από έναν κύβο στην αρχή

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(x, y, z);
glTranslatef(x1, y1, z1);
Draw_cube_origin();
glTranslatef(x2, y2, z2);
Draw_cube _origin();
….
```

- ΛΑΘΟΣ!!!

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
glTranslatef(x, y, z);
glPushMatrix(); //αποθήκευση στο stack
glTranslatef(x1, y1, z1);
Draw_cube_origin();
glPopMatrix();
glPushMatrix();
glTranslatef(x2, y2, z2);
Draw_cube _origin();
glPopMatrix();
….
```

Η θεώρηση των δύο πλαισίων

- Οι μετασχηματισμοί μετακινούν το κοσμικό πλαίσιο ως προς το πλαίσιο της κάμερας και εφαρμόζονται ως προς αυτό το μετακινούμενο πλαίσιο
- Οι μετασχηματισμοί εφαρμόζονται με τη σειρά με την οποία ορίζονται στο πρόγραμμα!
- Στο τέλος το αντικείμενο ζωγραφίζεται πάνω στο κοσμικό πλαίσιο
- Οι κλιμακώσεις συμπιέζουν/επιμηκύνουν την κλίμακα των αξόνων

Η θεώρηση των δύο πλαισίων

- Η ίδια σειρά εντολών παράγει το ΙΔΙΟ αποτέλεσμα στις δύο θεωρήσεις, η σκοπιά που βλέπουμε το πρόβλημα αλλάζει.
- Πρόβλημα με συνδυασμό μη ομοιόμορφων κλιμακώσεων και περιστροφών
- Ο πίνακας modelview μετασχηματίζει από συντεταγμένες κόσμου σε συντεταγμένες κάμερας.

Μετασχηματισμοί μοντελοποίησης-παρατήρησης

- Μέχρι τώρα είδαμε τους μετασχηματισμούς σαν τοποθέτηση του αντικειμένου στο χώρο με ακίνητη την κάμερα: μετασχηματισμοί μοντελοποίησης.
- Μπορούμε να μετακινήσουμε την κάμερα ως προς το κοσμικό σύστημα συντεταγμένων: μετασχηματισμοί παρατήρησης

Μετασχηματισμοί μοντελοποίησης-παρατήρησης

- Φυσικό ανάλογο: τοποθέτηση της κάμερας & τοποθέτηση των αντικειμένων.
- Μετασχηματισμοί μοντελοποίησης & παρατήρησης: δυικότητα
  - Μετακίνηση του αντικειμένου κατά d στον z με σταθερή κάμερα ισοδυναμεί με μετακίνηση της κάμερας κατά -d στον z με σταθερό αντικείμενο!

Μετασχηματισμοί μοντελοποίησης-παρατήρησης

- Πίνακας modelview: view*model
- Οι εντολές μετασχηματισμών μπορούν να ιδωθούν σαν εντολές μετασχηματισμού παρατήρησης: η κάμερα ως αντικείμενο.

Μετασχηματισμοί παρατήρησης

- Συνήθως οι εντολές που θεωρούμε ότι αφορούν την παρατήρηση (μετακινούν την κάμερα) μπαίνουν στην αρχή.
- Εντολές παρατήρησης & OpenGL(ένα πλαίσιο): Οι μετασχηματισμοί εφαρμόζονται με τη σειρά με την οποία ορίζονται στο πρόγραμμα με «αντίστροφα» ορίσματα.
- Μεικτή θεώρηση εντολών

Παράδειγμα

glTranslate3f(0, 0, -10);

glRotate3f(90, 0, 1, 0);

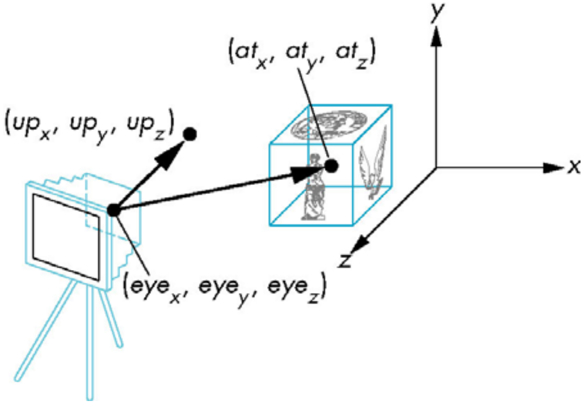
glScale3f(2, 2, 2);

glTranslate3f(10, 0, 0);

- Θεώρηση με τους τρεις τρόπους

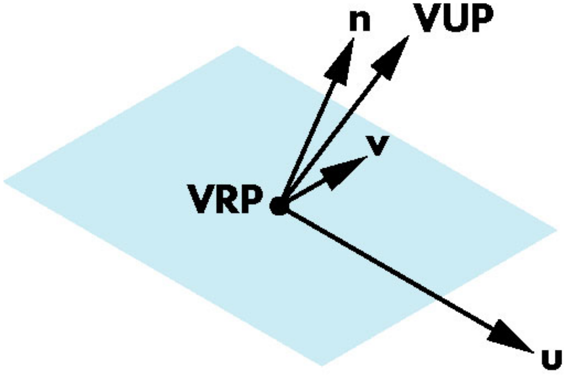
Εντολή παρατήρησης

- Δεν υπάρχει ανάγκη για διαχωρισμό εντολών μοντελοποίησης & παρατήρησης
- Μοναδική «καθαρή» εντολή παρατήρησης :
- gluLookAt(eyex, eyey, eyez, atx, aty, atz upx, upy, upz)
- Εύκολος χειρισμός κίνησης της κάμερας



Τοποθέτηση της κάμερας

- OpenGL: είτε με μετασχηματισμούς (πρέπει να βρεθούν οι κατάλληλες τιμές), είτε με την gluLookAt
- PHIGS: Προσδιορισμός της θέσης της κάμερας από:
  - View reference point (VRP)
  - View plane normal (VPN)
  - View up vector (VUP)



Τοποθέτηση της κάμερας

- Πλαίσιο συντεταγμένων παρατήρησης ή **u-v-n**
- **n**=VPN
- **v**: προβολή του VUP στο επίπεδο προβολής
- **u**=**v**x**n**
- **VRP**

Τοποθέτηση της κάμερας

- Χρήση σε OpenGL:
  - Καθορίζω τις συντεταγμένες των VRP, VUP, VPN στο κοσμικό σύστημα.
  - Βρίσκω τον πίνακα μετασχηματισμού (modelview) που θα δώσει την κατάλληλη θέση της κάμερας.
- VRP, VUP, VPN ⇨ **u,v,n**
- Αλλαγή πλαισίου.

Τοποθέτηση της κάμερας

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \mathbf{n} = \begin{bmatrix} n_x \\ n_y \\ n_z \\ 0 \end{bmatrix} \qquad \mathbf{v}_{up} = \begin{bmatrix} v_{upx} \\ v_{upy} \\ v_{upz} \\ 0 \end{bmatrix}$$

Τοποθέτηση της κάμερας

$$\mathbf{v} = \mathbf{v}_{up} - \frac{\mathbf{v}_{up} \cdot \mathbf{n}}{\mathbf{n} \cdot \mathbf{n}} \mathbf{n} \qquad \mathbf{u} = \mathbf{v} \times \mathbf{n}$$

- Κανονικοποίηση των **u, v, n**

$$\mathbf{V} = \begin{bmatrix} u'_x & u'_y & u'_z & -xu'_x - yu'_y - zu'_z \\ v'_x & v'_y & v'_z & -xv'_x - yv'_y - zv'_z \\ n'_x & n'_y & n'_z & -xn'_x - yn'_y - zn'_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Παράδειγμα

- Ισομετρική προβολή

$$\mathbf{p} = \begin{bmatrix} -d \\ d \\ d \\ 1 \end{bmatrix} \quad \mathbf{n} = \begin{bmatrix} 1 \\ -1 \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{v}_{up} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$