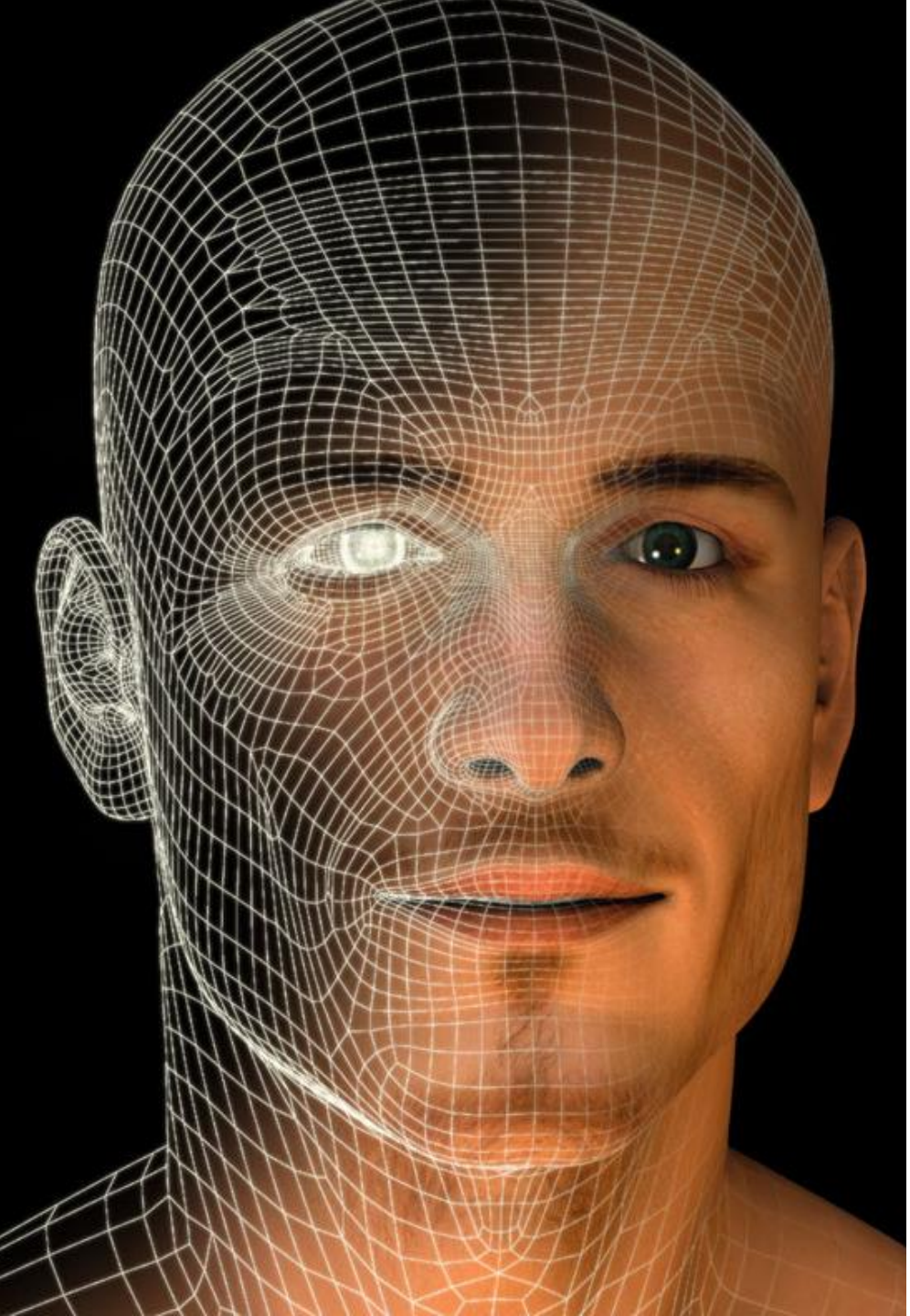


Τμήμα Μηχανικών Πληροφορικής

ΤΕΙ Ανατολικής Μακεδονίας και Θράκης

Γραφικά Υπολογιστών
ΣΤ’ Εξάμηνο

Δρ Κωνσταντίνος Δεμερτζής

ΤΕΙ Ανατολικής Μακεδονίας και Θράκης
Τμήμα Μηχανικών Πληροφορικής

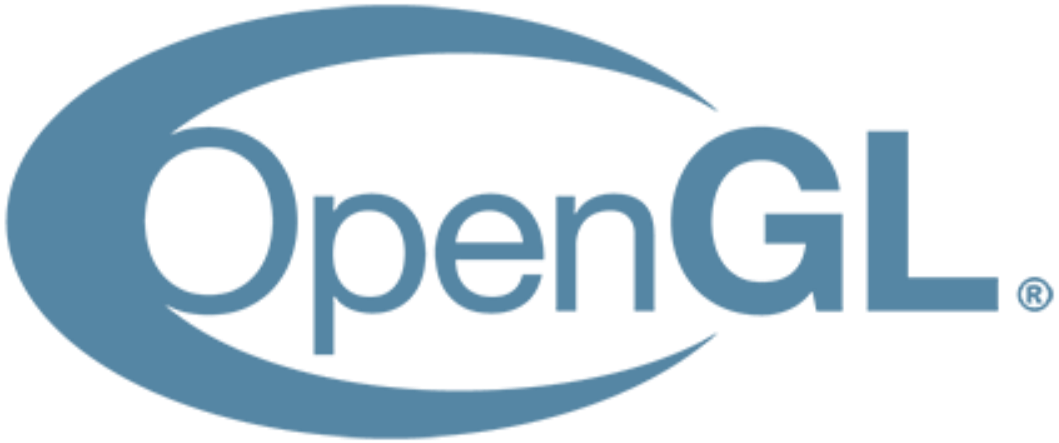
Γραφικά Υπολογιστών

1η Ενότητα

OpenGL

ΤΕΙ Ανατολικής Μακεδονίας και Θράκης
Τμήμα Μηχανικών Πληροφορικής

Γραφικά Υπολογιστών



Γραφικά Υπολογιστών

Τι είναι η OpenGL

- ✓ *Η OpenGL δεν είναι μια συγκεκριμένη βιβλιοθήκη γραφικών.*
- ✓ *Είναι ένα πρότυπο που καθορίζει τις λειτουργίες που πρέπει να υποστηρίζει μία βιβλιοθήκη γραφικών ούτως ώστε να είναι συμβατή με αυτήν.*
- ✓ *Η OpenGL ορίζει μια προγραμματιστική διεπιφάνεια (Application PrograMMing Interface) σχεδίασης γραφικών.*

Χαρακτηριστικά της OpenGL

- ✓ *Ο προγραμματιστής προγραμματίζει χρησιμοποιώντας μια υλοποίηση της OpenGL.*
- ✓ *Μια βιβλιοθήκη που υλοποιεί το πρότυπο της OpenGL μπορεί να συνταχθεί σε οποιαδήποτε γλώσσα προγραμματισμού (η OpenGL είναι πρότυπο ανεξάρτητο πλατφόρμας).*
- ✓ *Οι περισσότεροι μεταγλωττιστές εμπεριέχουν ή μπορεί να ενσωματωθεί σε αυτούς μία βιβλιοθήκη της OpenGL.*
- ✓ *Στο παρόν σεμινάριο θα χρησιμοποιηθεί υλοποίηση της OpenGL στη γλώσσα προγραμματισμού C.*

Γραφικά Υπολογιστών

Κατηγορίες βιβλιοθηκών της OpenGL

- ✓ ***GL Core Library***: *Περιέχει βασικές εντολές σχεδίασης (σχεδίαση βασικών γεωμετρικών σχημάτων, ορισμός χρωμάτων κλπ.) Ολες οι ρουτίνες της βιβλιοθήκης ξεκινούν με το πρόθεμα gl*
- ✓ ***OpenGL Utility Library (GLU)***: *Περιέχει πιο εξειδικευμένες λειτουργίες (Σχεδίαση σύνθετων επιφανειών, ορισμός προβολών κλπ) Ολες οι ρουτίνες της βιβλιοθήκης ξεκινούν με το πρόθεμα glu*
- ✓ ***OpenGL Utility Toolkit (GLUT)***: *Ρουτίνες εισόδου-εξόδου (σχηματισμός παραθύρων, διαχείριση γεγονότων κλπ.). Κατ' εξαίρεση, οι εντολές της GLUT εξαρτώνται από την αρχιτεκτονική του συστήματος. Ολες οι εντολές της βιβλιοθήκης ξεκινούν με το πρόθεμα glut.*

Γραφικά Υπολογιστών

Ένα τυπικό παράδειγμα

```
#include <glut.h>
void display()
{
  glClearColor(1,1,1,0);
  glClear(GL_COLOR_BUFFER_BIT);

  glBegin(GL_LINES);
  glColor3f(1,0,0);
  glVertex2i(20,20);
  glVertex2i(40,40);
  glEnd();
  glFlush();
}
```

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GL
UT_RGB);
glutCreateWindow("A saMple OpenGL
application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

Γραφικά Υπολογιστών

Κεφαλίδες της OpenGL

- ✓ ***#include <glut.h>=*** *Επιτρέπει τη χρήση βιβλιοθηκών της OpenGL*
- ✓ ***#include <gl.h>=*** *OpenGL Core Library*
- ✓ ***#include <glu.h>=*** *OpenGL Utility*
- ✓ ***#include <glut.h>=*** *OpenGL Core Library*
  + OpenGL Utility
  + OpenGL Utility Toolkit

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

glutInit(): Ενεργοποιεί τη χρήση των εντολών της βιβλιοθήκης GLUT.

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

glutInitWindowPosition:
Δηλώνει τη θέση στην οθόνη όπου θα εμφανιστεί το παράθυρο της εφαρμογής

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

glutInitWindowSize:
Ορίζει τις διαστάσεις του παραθύρου της εφαρμογής σε pixels

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

glutInitDisplayMode:
Καθορίζει ρυθμίσεις απεικόνισης (μοντέλο ενταμίευσης, χρωματικό μοντέλο κ.λ.π.)

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

glutCreateWindow: → glutCreateWindow("A saMple OpenGL application");

Εμφανίζει το παράθυρο της εφαρμογής

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

glMatrixMode: Επιλέγει το μητρώο που επιθυμούμε να τροποποιήσουμε (προβολής ή μετασχηματισμού μοντέλου)

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0 }
```

gluOrtho2D: Δήλωση παράλληλης προβολής

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

glutDisplayFunc:
Δηλώνει τη συνάρτηση που θα εκτελείται κάθε φορά που απαιτείται σχεδιασμός της σκηνής

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

```
int Main(int argc, char** argv){
glutInit(&argc,argv);
glutInitWindowPosition(50,50);
glutInitWindowSize(640,480);
glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
glutCreateWindow("A saMple OpenGL application");
glMatrixMode(GL_PROJECTION);
gluOrtho2D(0,50,0,50);
glutDisplayFunc(display);
glutMainLoop();
return 0;}
```

glutMainLoop:
Ενεργοποιεί τον κύκλο ακρόασης γεγονότων

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

glClearColor:
Δήλωση χρώματος καθαρισμού της οθόνης

```
void display()
{
 glClearColor(1,1,1,0);
 glClear(GL_COLOR_BUFFER_BIT);

 glColor3f(1,0,0);

 glBegin(GL_LINES);
 glVertex2i(20,20);
 glVertex2i(40,40);
 glEnd();
 glFlush();
 }
```

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

glClear: Καθαρισμός οθόνης (καθαρισμός ενός από τους ενταμιευτές του συστήματος γραφικών)

```
void display()
{
  glClearColor(1,1,1,0);
  glClear(GL_COLOR_BUFFER_BIT);

  glColor3f(1,0,0);

  glBegin(GL_LINES);
  glVertex2i(20,20);
  glVertex2i(40,40);
  glEnd();
  glFlush();
  }
```

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

glColor*: Επιλογή χρώματος σχεδίασης

```
void display()
{
  glClearColor(1,1,1,0);
  glClear(GL_COLOR_BUFFER_BIT);

  glColor3f(1,0,0);

  glBegin(GL_LINES);
  glVertex2i(20,20);
  glVertex2i(40,40);
  glEnd();
  glFlush();
  }
```

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

```
void display()
{
  glClearColor(1,1,1,0);
  glClear(GL_COLOR_BUFFER_BIT);

  glColor3f(1,0,0);

  glBegin(GL_LINES);
  glVertex2i(20,20);
  glVertex2i(40,40);
  glEnd();
  glFlush();
  }
```

glBegin/glEnd: Μεταξύ αυτών των εντολών δηλώνονται συντεταγμένες κορυφών γεωμετρικών σχημάτων. Το είδος των σχημάτων καθορίζεται από το όρισμα της glBegin.

Γραφικά Υπολογιστών

Ανάλυση παραδείγματος

```
void display()
{
  glClearColor(1,1,1,0);
  glClear(GL_COLOR_BUFFER_BIT);

  glColor3f(1,0,0);

  glBegin(GL_LINES);
  glVertex2i(20,20);
  glVertex2i(40,40);
  glEnd();
  glFlush();
  }
```

glVertex*: Δήλωση συντεταγμένων μεμονωμένων κορυφών

Γραφικά Υπολογιστών

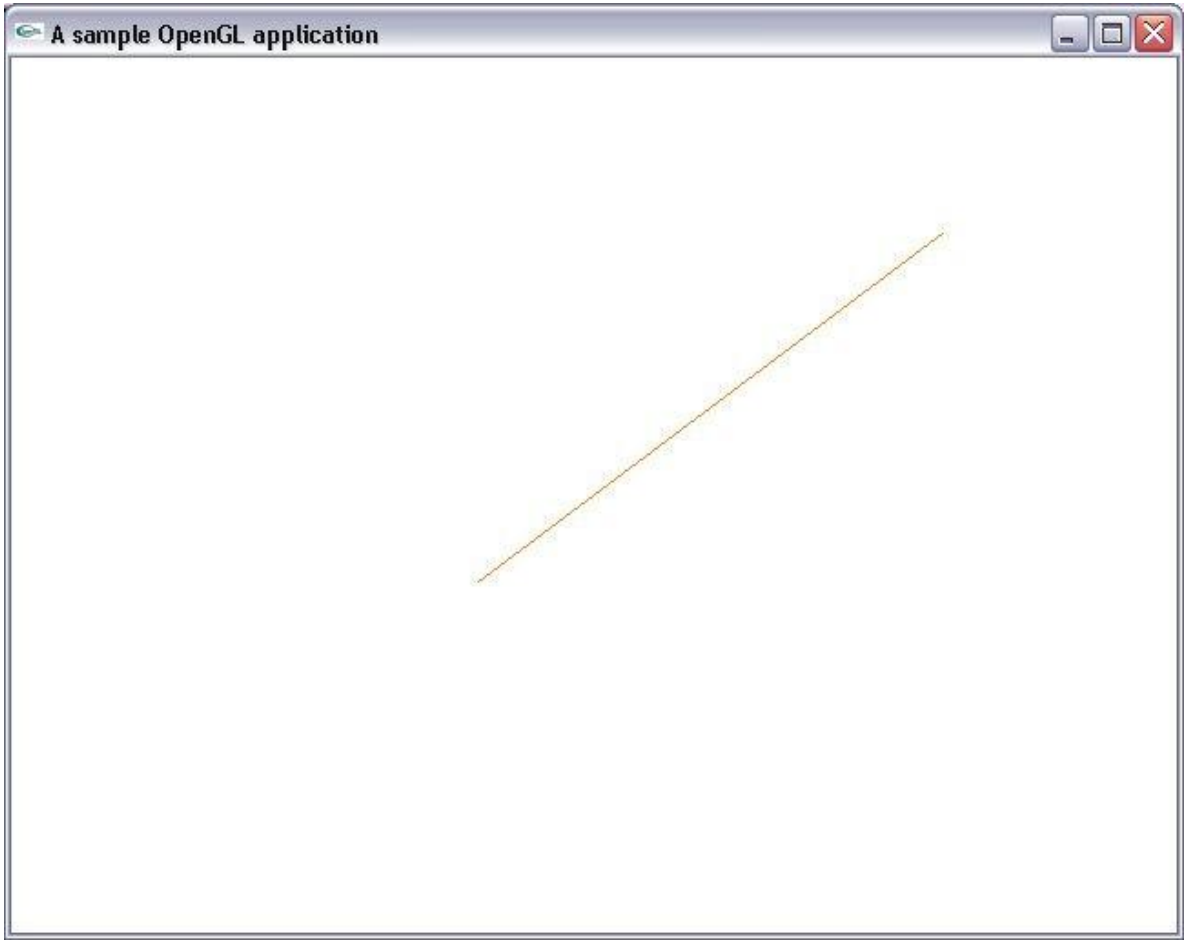
Ανάλυση παραδείγματος

```
void display()
{
  glClearColor(1,1,1,0);
  glClear(GL_COLOR_BUFFER_BIT);

  glColor3f(1,0,0);

  glBegin(GL_LINES);
  glVertex2i(20,20);
  glVertex2i(40,40);
  glEnd();
  glFlush();
}
```

glFlush: Προωθεί την εκτέλεση εντολών που εκκρεμούν.

Γραφικά Υπολογιστών

Αποτέλεσμα παραδείγματος



Γραφικά Υπολογιστών

Πρωτογενείς τύποι δεδομένων

Τύπος της OpenGL	Τύπος δεδομένων	Αντίστοιχος τύπος στη C	Επίθημα
GLbyte	ακέραιος 8bits	signed char	b
GLshort	ακέραιος 16bits	short	s
GLint / GLsizei	ακέραιος 32bits	int/long	i
GLfloat / GLclaMpf	κινητής υποδιαστολής	float	f
GLdouble / GLclaMpd	κινητής υποδιαστολής διπλής ακρίβειας	double	d
GLubyte / GLboolean	ακέραιος 8bits χωρίς πρόσημο	unsigned char	ub
GLushort	ακέραιος 16bits χωρίς πρόσημο	unsigned short	us
GLuint/GLenuM/GLbitfield	ακέραιος 32bits χωρίς πρόσημο	unsigned int unsigned long	ui

Γραφικά Υπολογιστών

Ονοματολογία συμβολικών σταθερών στην OpenGL

- ✓ ***Συμβολικές σταθερές:*** *προκαθορισμένες σταθερές που συνήθως χρησιμοποιούνται ως ορίσματα σε εντολές τις OpenGL και εκπροσωπούν συγκεκριμένες ρυθμίσεις.*
- ✓ ***Συμβολικές σταθερές της κύριας βιβλιοθήκης (core library):*** ***GL_(όνομα_σταθεράς)*****, πχ** ***GL_COLOR_BUFFER_BIT GL_LIGHTING***
- ✓ ***Συμβολικές σταθερές της της βιβλιοθήκης GLUT:*** ***GLUT_(όνομα_σταθεράς)*****, πχ** ***GLUT_RGB***

Γραφικά Υπολογιστών

Ονοματολογία εντολών στην OpenGL

- ✓ *Στην OpenGL για ορισμένες εντολές ορίζονται πολλαπλές παραλλαγές, ανάλογα με:*
  - ✓ *τον τύπο των ορισμάτων που δέχονται (π.χ. ακέραιοι ή πραγματικοί),*
  - ✓ *τις διαστάσεις του χώρου (π.χ. σχεδίαση σε δύο ή τρείς διαστάσεις)*
  - ✓ *το χρωματικό μοντέλο*
  - ✓ *τον τρόπο με τον οποίο περνάμε ορίσματα (call by value ή call by reference).*
- ✓ *Οι εντολές στην OpenGL έχουν επίθημα που καθορίζει το πλήθος και τύπο των ορισμάτων που δέχονται.*
- ✓ *Τα επιθήματα συνδυάζονται ως εξής:*

gl
+όνομα εντολής
+διάσταση χώρου ή πλήθος χρωματικών τιμών {2,3,4}
+επίθημα πρωτογενούς τύπου δεδομένων {ubsifd}
+είδος ορισμάτων (τιμές ή δείκτες σε μητρώα) {-/v}
glFunctionNaMe{234}{ubsifd}{v}

Γραφικά Υπολογιστών

Παράδειγμα ονοματολογίας: glVertex

- ✓ *Δήλωση σημείου στο διδιάστατο χώρο {2} με συντεταγμένες που δίνονται ως πραγματικοί αριθμοί απλής ακρίβειας {f}:*
  - ✓ *glVertex2f (GLfloat x, GLfloat y);*
- ✓ *Δήλωση σημείου στον τρισδιάστατο χώρο {3} με συντεταγμένες που δίνονται ως προσημασμένοι ακέραιοι {i}:*
  - ✓ *glVertex3i(GLint x, GLint y, GLint z);*
- ✓ *Για πραγματικές συντεταγμένες {f} ενός σημείου στον τρισδιάστατο χώρο {3} που δίνονται με τη μορφή μητρώου (όρισμα δείκτης σε πίνακα) {v}:*
  - ✓ *GLfloat coord[ ]={1,2,3}; glVertex3fv(const GLfloat *coord);*

Γραφικά Υπολογιστών

Η OpenGL ως μηχανή καταστάσεων

- ✓ *Το περιβάλλον της OpenGL λειτουργεί ως μια μηχανή καταστάσεων*
  - ✓ *Μηχανή καταστάσεων: Ένα περιβάλλον, το οποίο, σε κάθε χρονική στιγμή, λειτουργεί βάσει προκαθορισμένων ιδιοτήτων, των μεταβλητών κατάστασης (state variables).*
  - ✓ *Μεταβλητές κατάστασης: Έχουν μια προκαθορισμένη αρχική τιμή η οποία μπορεί να μεταβληθεί κατά την πορεία της εκτέλεσης του κώδικα από τον προγραμματιστή.*
  - ✓ *Οι τρέχουσες τιμές των μεταβλητών παραμένουν ενεργές.*

Παραδείγματα μεταβλητών κατάστασης

- ✓ *Ορισμένα παραδείγματα μεταβλητών κατάστασης:*
  - ✓ *τρέχον χρώμα σχεδίασης*
  - ✓ *πάχος γραμμών*
  - ✓ *χρώμα καθαρισμού της οθόνης (φόντου)*
- ✓ *Είναι σημαντικό ο προγραμματιστής να αρχικοποιεί τις μεταβλητές κατάστασης, όποτε αυτό είναι απαραίτητο και να παρακολουθεί τις τιμές τους.*

Γραφικά Υπολογιστών

Ιδιότητες δύο καταστάσεων

- ✓ *Έχουν δύο πιθανές τιμές (TRUE ή FALSE)*
- ✓ *Καθορίζουν την ενεργοποίηση ή μη εξειδικευμένων λειτουργιών (μίξη χρωμάτων, φωτορεαλισμός, απόδοση υφής κ.λ.π).*
- ✓ *Δίνοντας στον προγραμματιστή τη δυνατότητα ενεργοποίησης λειτουργιών, η μηχανή της OpenGL βελτιστοποιεί το υπολογιστικό φορτίο.*
- ✓ *Οι υποστηριζόμενες λειτουργίες της μηχανής καταστάσεων της OpenGL ενεργοποιούνται και απενεργοποιούνται με τις εντολές (glEnable και glDisable αντίστοιχα):*
  - ✓ *void glEnable(GLenuM cap);*
  - ✓ *void glDisable(GlenuM cap);*
- ✓ *cap: Η ιδιότητα που ενεργοποιείται η απενεργοποιείται πχ glEnable(GL_BLEND); Ενεργοποίηση μίξης χρωμάτων*
- ✓ *Κάθε ιδιότητα που ενεργοποιείται παραμένει ενεργή σε όλη τη διάρκεια εκτέλεσης του προγράμματος*

Γραφικά Υπολογιστών

Επισκόπηση ιδιοτήτων δύο καταστάσεων

- ✓ *Προκειμένου να ελέγξουμε αν μια ιδιότητα δύο καταστάσεων είναι ενεργοποιημένη ή απενεργοποιημένη χρησιμοποιούμε την εντολή glIsEnabled:*
  - ✓ *GLboolean glIsEnabled(GLenuM capability);*
- ✓ *Επιστρέφει GL_TRUE ή GL_FALSE, ανάλογα με το ανη εξεταζόμενη ιδιότητα είναι ενεργοποιημένη ή όχι.*

Ιδιότητες κατάστασης πολλαπλών τιμών

- ✓ *Υπάρχουν μεταβλητές κατάστασης που μπορούν να πάρουν περισσότερες από δύο τιμές (πχ τρέχον χρώμα σχεδίασης ή πάχος γραμμών)*
- ✓ *Οι τιμές των συνθετων ιδιοτήτων κατάστασης ρυθμίζονται με ξεχωριστές εντολές της OpenGL η καθεμιά.*

Γραφικά Υπολογιστών

Επισκόπηση σύνθετων ιδιοτήτων κατάστασης

- ✓ *Οι εντολές glGet{Type}v χρησιμοποιούνται για την επισκόπηση της τρέχουσας τιμής μεταβλητών κατάστασης*
  - ✓ *void glGetBooleanv(GLenuM paraMeterNaMe, GLboolean *paraMeters); void glGetIntegerv(GLenuM paraMeterNaMe, GLint *paraMeters);*
  - ✓ *void glGetFloatv(GLenuM paraMeterNaMe, GLfloat *paraMeters); void glGetDoublev(GLenuM paraMeterNaMe, GLdouble *paraMeters);*
- ✓ *paraMeterNaMe: συμβολική σταθερά που καθορίζει την ιδιότητα*
- ✓ *paraMeters: δείκτης σε μητρώο όπου αποθηκεύεται η τιμή ή το σύνολο τιμών που προσδιορίζουν την ιδιότητα.*
- ✓ *Παράδειγμα:*
  - ✓ *Το μητρώο paraMeters θα περιέχει τις τιμές τριών χρωματικών συνιστωσών (μοντέλο RGB, διάσταση 3): GLfloat colorVal[3];*
  - ✓ *Παράμετρος = τρέχον χρώμα σχεδίασης: paraMeterNaMe = GL_CURRENT_COLOR*
  - ✓ *Αποθήκευση των τρεχουσών χρωματικών συνιστωσών στο μητρώο colorVal: glGetFloatv(GL_CURRENT_COLOR, colorVal);*

Γραφικά Υπολογιστών

Σχεδίαση στην OpenGL

- ✓ *Όλα τα γεωμετρικά σχήματα ορίζονται με δήλωση των κορυφών τους*
- ✓ *Σύνθετα σχήματα προσεγγίζονται από στοιχειώδη βασικά σχήματα.*
- ✓ *Κάθε σημείο της σκηνής αναπαρίσταται, στη γενική περίπτωση, σε τρισδιάστατο καρτεσιανό σύστημα συντεταγμένων.*
- ✓ *Το σύστημα συντεταγμένων είναι δεξιόστροφο.*

Ομογενείς συντεταγμένες

- ✓ *Όλα τα σημεία αναπαρίστανται με τέσσερις συντεταγμένες κινητής υποδιαστολής (x, y, z, w).*
- ✓ *Η δήλωση ενός σημείου με ομογενείς συντεταγμένες (x, y, z, w) ορίζει στον τρισδιάστατο χώρο χώρο ένα σημείο με συντεταγμένες (x/w, y/w, z/w)*
- ✓ *Αν δε δίνεται η συντεταγμένη z, το σύστημα θεωρεί την τιμή 0.*
- ✓ *Αν δε δίνεται η συντεταγμένη w, το σύστημα θεωρεί την τιμή 1.*

Γραφικά Υπολογιστών

Καθαρισμός οθόνης

- ✓ *Πριν το σχεδιασμό μιας σκηνής, απαιτείται ο καθαρισμός του ενταμιευτή χρωματικών τιμών (color buffer) του υπολογιστή,*
- ✓ *Ενταμιευτής χρωματικών τιμών: περιοχή μνήμης όπου αποθηκεύονται οι χρωματικές πληροφορίες για τη σχεδιαζόμενη σκηνή*
- ✓ *"Καθαρισμός" οθόνης = αρχικοποίηση των ενταμιευτών με κάποια προκαθορισμένη τιμή.*
- ✓ *Ο καθαρισμός της οθόνης γίνεται με το χρώμα φόντου που επιλέγει ο προγραμματιστής.*

Ρύθμιση χρώματος καθαρισμού

- ✓ *Το χρώμα καθαρισμού της οθόνης είναι μεταβλητή κατάστασης και ορίζεται με την εντολή glClearColor, πχ glClearColor(GLfloat red, GLfloat green, GLfloat blue, GLfloat alpha); (red, green, blue, alpha: τα βάρη του χρώματος στο χρωματικό μοντέλο RGBA)*
- ✓ *Το χρώμα καθαρισμού, ως μεταβλητή κατάστασης, διατηρεί την τελευταία τιμή που του ανατέθηκε.*

Γραφικά Υπολογιστών

Καθαρισμός buffer

- ✓ *Ο καθαρισμός buffer αυτός καθεαυτός γίνεται με την εντολή glClear:*
- ✓ *void glClear(GLEnuM buffer);*
- ✓ *Η μηχανή της OpenGL περιέχει πολλούς buffer, οπότε πρεπει να καθορίσουμε το είδος του ενταμιευτή που επιθυμούμε να καθαρίσουμε:*
  - ✓ *GL_COLOR_BUFFER_BIT: καθαρισμός ενταμιευτή χρωματικών τιμών (colour buffer)*
  - ✓ *GL_DEPTH_BUFFER_BIT: καθαρισμός ενταμιευτή βάθους (depth buffer)*
- ✓ *Παράδειγμα:*
  - ✓ *Καθαρισμός της οθόνης με μαύρο χρώμα*
  - ✓ *glClearColor(0.0, 0.0, 0.0, 0.0); glClear(GL_COLOR_BUFFER_BIT);*
  - ✓ *Ορίζουμε το χρώμα καθαρισμού στην αρχή του προγράμματός μας και κατόπιν καθαρίζουμε τους buffer όσο συχνά χρειάζεται.*

Γραφικά Υπολογιστών

Καθαρισμός πολλαπλών buffer

- ✓ *Η εντολή glClear επιτρέπει επίσης τον καθορισμό πολλαπλών ενταμιευτών με μία μόνο κλήση της.*
- ✓ *Δίνουμε ως παραμέτρους πολλαπλούς ενταμιευτές διαχωρισμένους με τελεστές OR (|).*
- ✓ *Π.χ. για τον ταυτόχρονο καθαρισμό του ενταμιευτή χρωματικών τιμών (color buffer) και του ενταμιευτή τιμών βάθους (depth buffer) δίνουμε:*
  - ✓ *glClearColor(0.0, 0.0, 0.0, 0.0);*
  - ✓ *glClearDepth(0.0);*
  - ✓ *glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);*

Καθορισμός χρωμάτων

- ✓ *Κάθε φορά που δίνουμε εντολή σχεδιάσης ενός συγκεκριμένου γεωμετρικού σχήματος, του αποδίδουμε το τρέχον χρώμα σχεδίασης*
- ✓ *Το χρώμα σχεδίασης, ως μεταβλητή κατάστασης, διατηρεί την τελευταία τιμή του.*
- ✓ *Για να καθορίσουμε ένα χρώμα, χρησιμοποιούμε την εντολή glColor3{ub,f,d}.*

Γραφικά Υπολογιστών

Καθορισμός χρωμάτων

- ✓ *Παράδειγμα:*
  - ✓ *void glColor3ub(GLubyte red, GLubyte green, GLubyte blue);* *0<r,g,b<255*
  - ✓ *void glColor3f(GLfloat red, GLfloat green, GLfloat blue);* *0<r,g,b<1*
  - ✓ *void glColor3d(GLdouble red, GLdouble green, GLdouble blue);* *0<r,g,b<1*

	RGB			CMYK				HLS			HSV			Gray
	Red	Green	Blue	Cyan	Magenta	Yellow	Black	Hue, lightness, saturation			Hue, saturation, value			Value
Red	255	0	0	0	100	100	0	0	50	100	0	100	100	-
Orange	255	128	0	0	50	100	0	30	50	100	30	100	100	-
Yellow	255	255	0	0	0	100	0	60	50	100	60	100	100	-
Bright green	0	255	0	100	0	100	0	120	50	100	120	100	100	-
Cyan	0	255	255	100	0	0	0	180	50	100	180	100	100	-
Blue	0	0	255	100	100	0	0	240	50	100	240	100	100	-
Violet	128	0	255	50	100	0	0	270	50	100	270	100	100	-
Magenta	255	0	255	0	100	0	0	300	50	100	300	100	100	-
White	255	255	255	0	0	0	0	NA	100	NA	NA	0	100	0
Mid-gray	128	128	128	0	0	0	50	NA	50	NA	NA	0	50	50
Black	0	0	0	0	0	0	100	NA	0	NA	NA	NA	0	100

Γραφικά Υπολογιστών

Καθορισμός κορυφών

- ✓ *Στην OpenGL, όλα τα γεωμετρικά σχήματα περιγράφονται δηλώνοντας τις κορυφές τους.*
- ✓ *Για τον καθορισμό κάθε μίας κορυφής χρησιμοποιούμε την εντολή glVertex*.*
  - ✓ *void glVertex{234}{sifd}[v](TYPE coords);*
- ✓ *Με τις παραλλαγές της εντολής glVertex, μπορούμε να δώσουμε από δύο (x, y) μέχρι και τέσσερις συντεταγμένες (x, y, z, w) για μία κορυφή.*
- ✓ *Παραδείγματα:*
  - ✓ *glVertex2s(2,3); Δήλωση σημείου με ακέραιες συντεταγμένες (x,y)=(2,3)*
  - ✓ *glVertex3d (0,0, 3.14); Δήλωση σημείου με συντεταγμένες (x,y,z)=(0,0,3.14)*
  - ✓ *glVertex4f (2.3, 1.0, -2.2, 2); Δήλωση σημείου (1.15, 0.5, -1.1) σε ομογενείς συντεταγμένες (w=2)*
  - ✓ *GLdouble dvect[3] = {5,9,1};*
  - ✓ *glVertex3dv(dvect); (Δήλωση σημείου που οι τιμές του ορίζονται στο μητρώο dvect)*

Γραφικά Υπολογιστών

Η δομή glBegin/glEnd

- ✓ *Ο ορισμός των κορυφών κάθε γεωμετρικού σχήματος περικλείεται μεταξύ δύο εντολών, των glBegin και glEnd.*
  - ✓ *void glBegin(GLEnuM Mode);*
  - ✓ *void glEnd();*
- ✓ *Μεταξύ των εντολών glBegin και glEnd περικλείονται εντολές δήλωσης σημείων ορίζουν ένα γεωμετρικό σχήμα ή μία ομάδα γεωμετρικών σχημάτων.*
- ✓ *Το είδος του γεωμετρικού σχήματος που ορίζεται, εξαρτάται από την συμβολική σταθερά Mode που δίνουμε ως παράμετρο στην εντολή glBegin.*

Γραφικά Υπολογιστών

Σχεδίαση μεμονωμένων σημείων (GL_POINTS)

- ✓ *Δίνουμε ως όρισμα στην glBegin τη σταθερά GL_POINTS.*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_POINTS);*
  - ✓ *glVertex2i(10,10);*
  - ✓ *glVertex2i(20,10);*
  - ✓ *glEnd();*
  - ✓ *Ορισμός δύο σημείων με συντεταγμένες (10,10) και (20,10).*
- ✓ *Για τη ρύθμιση του μεγέθους ενός σημείου, χρησιμοποιούμε την εντολή glPointSize:*
  - ✓ *void glPointSize(GLfloat size); (size: το πλάτος του σημείου σε pixels)*
- ✓ *Η προκαθορισμένη τιμή του πάχπυς γραμμής είναι 1.*
- ✓ *Το πάχος σημείων είναι μεταβλητή κατάστασης, άρα διατηρεί την τελευταία του τιμή.*

Γραφικά Υπολογιστών

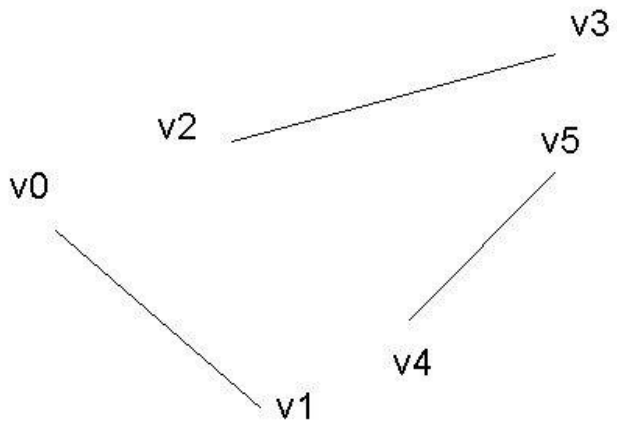
Σχεδίαση γραμμών

- ✓ *Οι γραμμές χαράσσονται ορίζοντας τις συντεταγμένες των άκρων τους.*
- ✓ *Οι εντολές σχεδιάσης γραμμών μας επιτρέπουν να σχεδιάσουμε:*
  - ✓ *α) Ευθύγραμμα τμήματα (GL_LINES)*
  - ✓ *β) Αλυσίδες ευθυγράμμων τμημάτων (GL_LINE_STRIP)*
  - ✓ *γ) Βρόχους ευθυγράμμων τμημάτων (GL_LINE_LOOP)*

Γραφικά Υπολογιστών

Σχεδίαση γραμμών

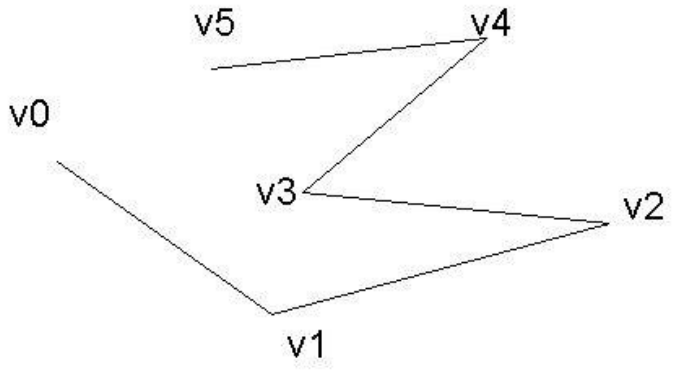
- ✓ *Ευθύγραμμα τμήματα (GL_LINES)*
  - ✓ *glBegin(GL_LINES);*
- ✓ *Τα δοθέντα σημεία ορίζουν ανά ζεύγη ευθύγραμμα τμήματα.*
- ✓ *Όταν ο αριθμός των σημείων είναι περιττός, το τελευταίο σημείο αγνοείται.*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_LINES);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glVertex*(v5);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

Σχεδίαση γραμμών

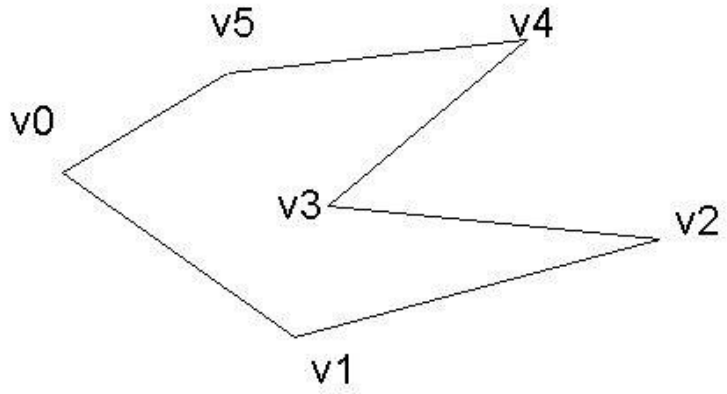
- ✓ *Αλυσίδες ευθυγράμμων τμημάτων (GL_LINE_STRIP)*
  - ✓ *glBegin(GL_LINE_STRIP);*
- ✓ *Τα σημεία ορίζουν διαδοχικά ευθύγραμμα τμήματα.*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_LINE_STRIP);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glVertex*(v5);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

Σχεδίαση γραμμών

- ✓ *Βρόχους ευθυγράμμων τμημάτων (GL_LINE_LOOP)*
  - ✓ *glBegin(GL_LINE_LOOP):*
- ✓ *Τα σημεία ορίζουν διαδοχικά ευθύγραμμα τμήματα.*
- ✓ *Το τελευταίο σημείο ενώνεται με το αρχικό.*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_LINE_LOOP);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glVertex*(v5);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

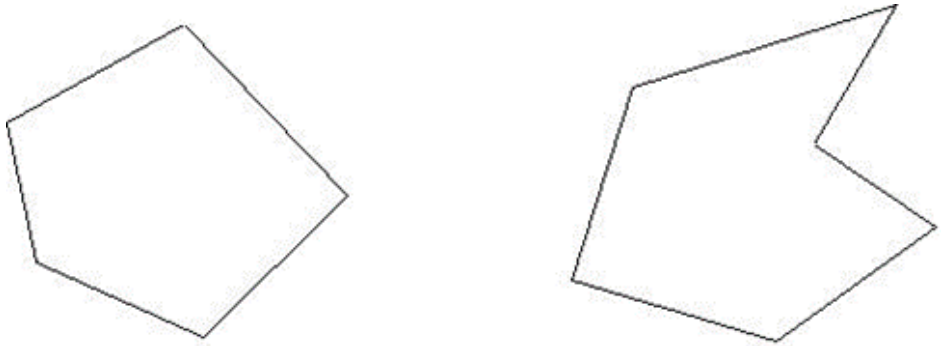
Ιδιότητες γραμμών

- ✓ *Πάχος γραμμών*
  - ✓ *Τροποποιείται με τη χρήση της εντολής glLineWidth:*
  - ✓ *void glLineWidth(GLfloat width);*
  - ✓ *width: το πάχος της γραμμής σε pixels*
- ✓ *Διάστιξη γραμμών*
  - ✓ *Χρήσιμη για τη σχεδιάση διακεκομμένων γραμμών*
  - ✓ *Η διάστιξη γραμμών ενεργοποιείται δίνοντας την παράμετρο GL_LINE_STIPPLE στην εντολή glEnable( ):*
  - ✓ *glEnable(GL_LINE_STIPPLE);*

Γραφικά Υπολογιστών

Πολύγωνα

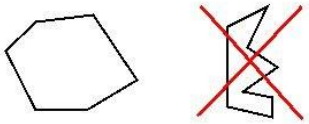
- ✓ *Πολύγωνα: περιοχές που περικλείονται από βρόχους ευθύγραμμων τμημάτων.*
- ✓ *Τα πολύγωνα σχεδιάζονται συμπαγή ή τα περιγράμματά τους ή οι κορυφές τους.*
- ✓ *Κατηγορίες πολυγώνων: κυρτά και κοίλα*
  - ✓ *Κυρτά πολύγωνα: Όλες οι εσωτερικές γωνίες τους είναι μικρότερες των 180 μοιρών*
  - ✓ *Κοίλα πολύγωνα: Περιέχουν τουλάχιστον μια εσωτερική γωνία μεγαλύτερη των 180 μοιρών.*



Γραφικά Υπολογιστών

Περιορισμοί στη σχεδίαση πολυγώνων

- ✓ *α) Οι ρουτίνες της OpenGL σχεδιάζουν κυρτά πολύγωνα.*



- ✓ *β) Οι πλευρές των πολυγώνων δεν μπορούν να τέμνονται.*

- ✓ *γ) Οι ρουτίνες της OpenGL δε μπορούν να σχεδιάσουν πολύγωνα με οπές.*

- ✓ *Εάν δώσουμε εντολή σχεδίασης ενός "μη έγκυρου πολυγώνου", το αποτέλεσμα θα είναι απρόβλεπτο.*

Γραφικά Υπολογιστών

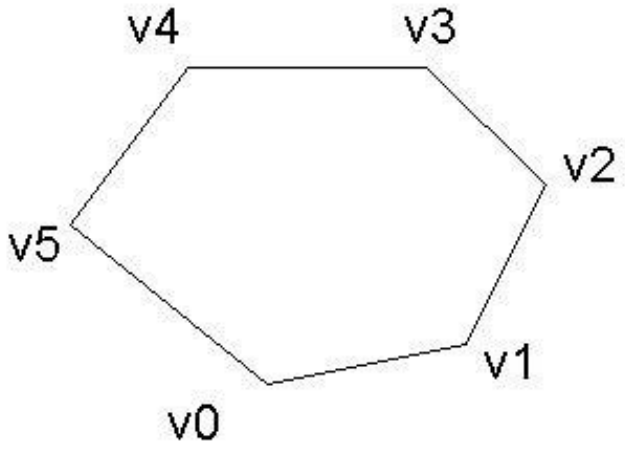
Σχεδίαση Πολύγωνων

- ✓ *Δηλώνουμε τις κορυφές ενός ή περισσοτέρων πολυγώνων μεταξύ των εντολών glBegin και glEnd.*
- ✓ *Τρεις καταστάσης σχεδίασης:*
  - ✓ *α) Σχεδίαση πολύγωνων (GL_POLYGON)*
  - ✓ *β) Σχεδίαση τετραπλεύρων (GL_QUADS)*
  - ✓ *γ) Σχεδίαση αλυσίδας τετραπλεύρων (GL_QUAD_STRIP)*

Γραφικά Υπολογιστών

Σχεδίαση Πολύγωνων

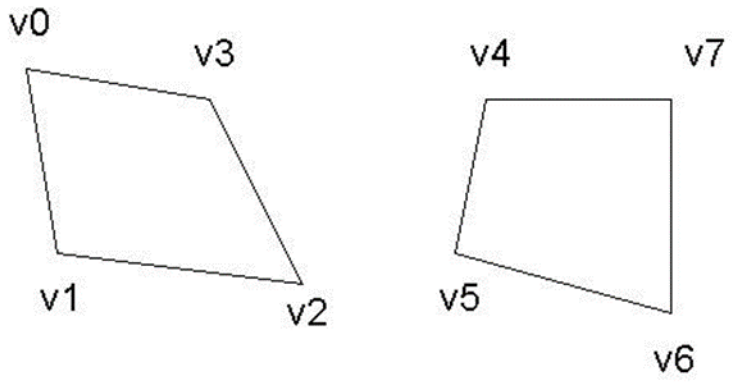
- ✓ *Σχεδίαση πολύγωνων (GL_POLYGON)*
  - ✓ *glBegin(GL_POLYGON);*
- ✓ *Τα σημεία ορίζουν τις διαδοχικές κορυφές ενός μόνο πολυγώνου (πλήθος κορυφών ≥ 3)*
- ✓ *Το πολύγωνο πρέπει να είναι κυρτό και να μην έχει τεμνόμενες πλευρές.*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_POLYGON);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glVertex*(v5);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

Σχεδίαση Πολύγωνων

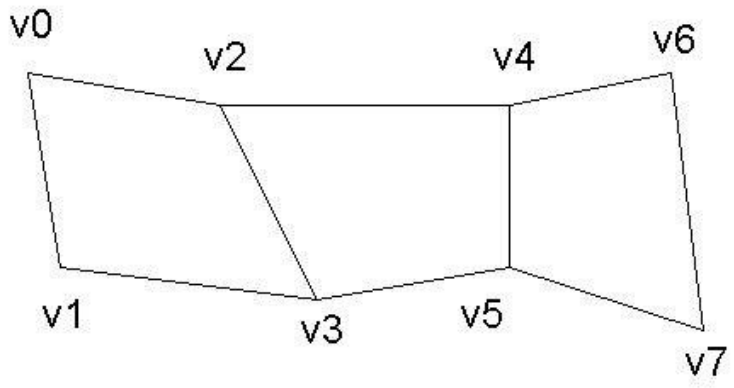
- ✓ *Σχεδίαση τετραπλεύρων (GL_QUADS)*
  - ✓ *glBegin(GL_QUADS):*
- ✓ *Οι κορυφές ορίζουν ανά τέσσερις διακεκριμένα τετράπλευρα.*
  - ✓ *(v0, v1,v2,…,vn-1)*
  - ✓ *(v0,v1,v2,v3),*
  - ✓ *(v4,v5,v6,v7) κοκ*
- ✓ *Αν το πλήθος κορυφών δεν είναι πολλαπλάσιο του 4, τότε η μία, δύο ή τρεις τελευταίες κορυφές παραλείπονται.*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_QUADS);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glVertex*(v5);*
  - ✓ *glVertex*(v6);*
  - ✓ *glVertex*(v7);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

Σχεδίαση Πολύγωνων

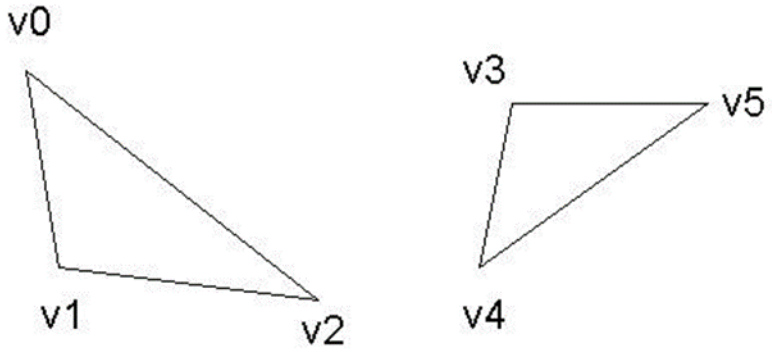
- ✓ *Σχεδίαση αλυσίδας τετραπλεύρων (GL_QUAD_STRIP)*
  - ✓ *glBegin(GL_QUAD_STRIP);*
- ✓ *Ορισμός αλυσίδας τετραπλεύρων με μία κοινη πλευρά.*
  - ✓ *(v0,v1,v2,…,vn)*
  - ✓ *(v0,v1,v3,v2),*
  - ✓ *(v2,v3,v5,v4)*
  - ✓ *(v4,v5,v7,v6) κοκ*
- ✓ *Εάν το πλήθος των κορυφών είναι περιττό, η τελευταία κορυφή παραλείπεται.*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_QUAD_STRIP);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glVertex*(v5);*
  - ✓ *glVertex*(v6);*
  - ✓ *glVertex*(v7);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

Σχεδίαση Τριγώνων

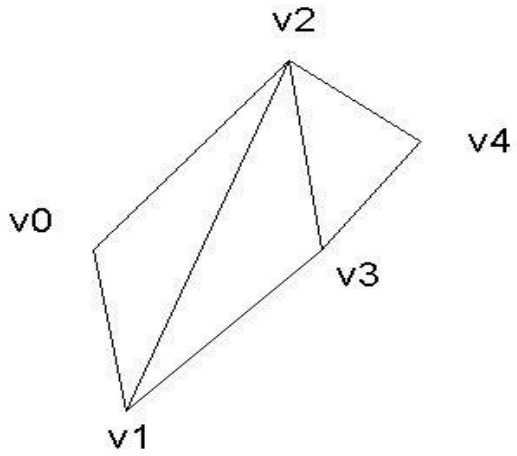
- ✓ *Σχεδίαση ανεξαρτήτων τριγώνων (GL_TRIANGLES)*
  - ✓ *glBegin(GL_TRIANGLES);*
- ✓ *Τα σημεία ορίζουν ανά τριάδες ανεξάρτητα τρίγωνα.*
  - ✓ *(v0,v1,v2,…,vn-1)*
  - ✓ *(v0,v1,v2),*
  - ✓ *(v3,v4,v5) κοκ*
- ✓ *Αν το πλήθος κορυφών δεν είναι ακέραιο πολλαπλάσιο του 3, η τελευταία ή οι δύο τελευταίες κορυφές παραλείπονται.*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_TRIANLGES);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glVertex*(v5);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

Σχεδίαση Τριγώνων

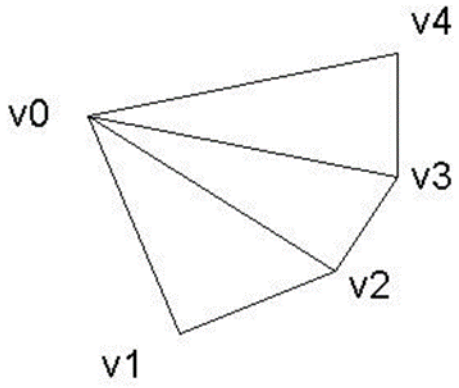
- ✓ *Σχεδίαση αλυσίδας τριγώνων με κοινή πλευρά (GL_TRIANGLE_STRIP)*
- ✓ *glBegin(GL_TRIANGLE_STRIP):*
- ✓ *Οι κορυφές ορίζουν μια αλυσίδα τριγώνων.*
- ✓ *Διαδοχικά τρίγωνα της αλυσίδας έχουν μία κοινή πλευρά.*
  - ✓ *(v0,v1,v2,…,vn)*
  - ✓ *(v0,v1,v2),*
  - ✓ *(v2,v1,v3),*
  - ✓ *(v2,v3,v4) κοκ*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_TRIANLGE_STRIP);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

Σχεδίαση Τριγώνων

- ✓ *Σχεδίαση αλυσίδας τριγώνων με κοινή κορυφή (GL_TRIANGLE_FAN)*
  - ✓ *glBegin(GL_TRIANGLE_FAN):*
- ✓ *Οι κορυφές ορίζουν μια αλυσίδα τριγώνων.*
- ✓ *Διαδοχικά τρίγωνα της αλυσίδας έχουν μία κοινή πλευρά και όλα τα τρίγωνα έχουν την πρώτη κορυφή κοινή*
  - ✓ *(v0,v1,v2,…,vn)*
  - ✓ *(v0,v1,v2),*
  - ✓ *(v0,v2,v3),*
  - ✓ *(v0,v3,v4) κοκ*
- ✓ *Παράδειγμα:*
  - ✓ *glBegin(GL_TRIANGLE_FAN);*
  - ✓ *glVertex*(v0);*
  - ✓ *glVertex*(v1);*
  - ✓ *glVertex*(v2);*
  - ✓ *glVertex*(v3);*
  - ✓ *glVertex*(v4);*
  - ✓ *glEnd();*



Γραφικά Υπολογιστών

Σχεδίαση ορθογωνίων

- ✓ *Η OpenGL επιτρέπει το σχεδιασμό ορθογωνίων με την εντολή glRect*.*
  - ✓ *void glRect{sifd}(TYPE x1, TYPE y1, TYPE x2, TYPE y2);*
  - ✓ *TYPE: τύπος δεδομένος που καθορίζεται από την παραλλαγή της*
  - ✓ *glRect* που επιλέγουμε (GLfloat, GLint κλπ)*
  - ✓ *(x1,y1) (x2,y2): συντεταγμένες των κορυφών της μίας διαγωνίου του ορθογωνίου*
- ✓ *Χρησιμοποιώντας την glRect τα ορθογώνια σχεδιάζονται στο επίπεδο z=0 με τις ακμές τους παράλληλες στους άξονες x,y.*
- ✓ *Οι παραλλαγές glRect*v δέχονται τις συντεταγμένες των κορυφών της διαγωνίου με τη μορφή μητρώων*
  - ✓ *void glRect{sifd}v(TYPE *v1, TYPE *v2);*

Γραφικά Υπολογιστών

Σχεδίαση καμπυλών

- ✓ *Οποιαδήποτε καμπύλη γραμμή μπορεί να προσεγγιστεί από στοιχειώδη ευθύγραμμα τμήματα.*
- ✓ *Οποιαδήποτε επιφάνεια μπορεί να προσεγγιστεί ή από ένα πολυγωνικό πλέγμα.*
- ✓ *Με επαρκή δειγματοληψία, μπορούμε να υποδιαιρέσουμε καμπύλες γραμμές και επιφάνειες σε επιμέρους ευθύγραμμα τμήματα ή επίπεδα πολύγωνα.*
- ✓ *Οι βιβλιοθήκες GLU και GLUT προσφέρουν εντολές για τη σχεδίαση ορισμένων σύνθετων γεωμετρικών σχημάτων.*
- ✓ *Η σχεδίαση κυκλικών σχημάτων επιτυγχάνεται χρησιμοποιώντας την παραμετρική εξίσωση κύκλου σε πολικές συντεταγμένες.*
- ✓ *Με ένα στοιχειώδες γωνιακό βήμα dθ, σχεδιάζουμε κυκλικά σχήματα, συνενώνοντας σημεία της περιφέρειάς του με στοιχειώδη ευθύγραμμα τμήματα.*
- ✓ *Για τον υπολογισμό των συντεταγμένων κάθε σημείου του κύκλου απαιτείται ο υπολογισμός δύο τριγωνομετρικών αριθμών, (επιπλέον υπολογιστικό κόστος).*
- ✓ *Συμμετρίες ως προς την ευθεία y=x και ως προς τους άξονες x,y.*
- ✓ *Μπορούμε να περιορίσουμε τη χρήση των παραμετρικών εξισώσεων στο ένα όγδοο του κύκλου.*
- ✓ *Αναπαραγάγουμε τα υπόλοιπα σημεία με σχέσεις συμμετρίας.*

Γραφικά Υπολογιστών

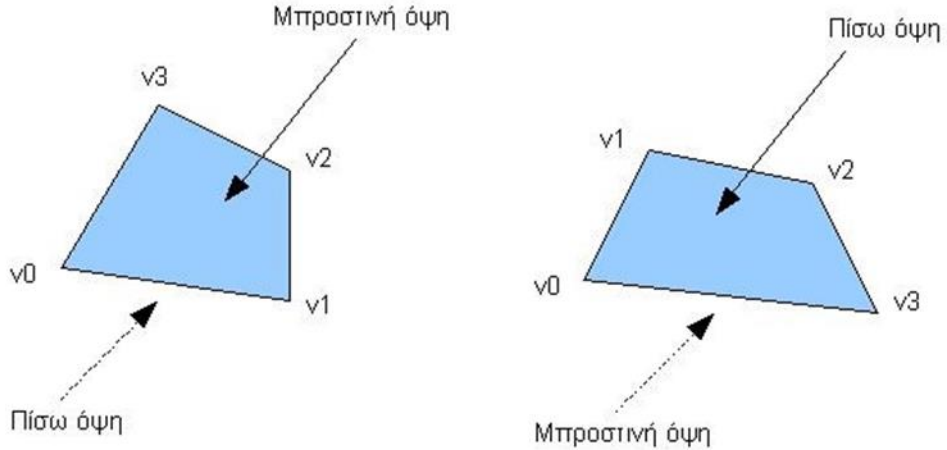
Προώθηση εντολών προς εκτέλεση (flushing)

- ✓ *Σε ένα υπολογιστικό σύστημα, οι εντολές εκτελούνται κατά ομάδες, αφού ολοκληρωθεί η συλλογή ενός πλήθους εντολών.*
- ✓ *Ωστόσο, για να σχεδιάστει ένα καρέ, ο προγραμματιστής θα πρέπει να είναι βέβαιος ότι, όσες εντολές έχουν δηλωθεί, προωθούνται προς εκτέλεση.*
- ✓ *Στην OpenGL ια να προωθήσουμε την εκτέλεση εντολών που εκκρεμούν, χρησιμοποιούμε την εντολής glFlush.*
  - ✓ *void glFlush( );*
- ✓ *Η εντολή glFlush πρέπει να εκτελείται κάθε φορά που ολοκληρώνουμε την περιγραφή του σκηνικού. Την τοποθετούμε στο τέλος της συνάρτησης που περιέχει τις εντολές σχεδίασης (display).*

Γραφικά Υπολογιστών

Όψεις πολυγώνων

- ✓ *Κάθε πολύγωνο χαρακτηρίζεται από δύο όψεις: τη μπροστινή και την πίσω όψη*
- ✓ *Η επιλογή του ποια όψη θα χαρακτηριστεί ως μπροστινή ή πίσω είναι αυθαίρετη.*
- ✓ *Η διάκριση αυτή είναι χρήσιμη στην περίπτωση που θέλουμε να αποδώσουμε διαφορετικά χαρακτηριστικά σε κάθε όψη μιας επιφάνειας (π.χ. διαφορετική υφή).*
- ✓ *Ο προσανατολισμός της επιφάνειας ενός πολυγώνου καθορίζεται ανάλογα με τη φορά δήλωσης των κορυφών του:*
  - ✓ *Εάν ο θεατής δηλώσει τις κορυφές του πολυγώνου με αριστερόστροφη φορά από τη δική του οπτική τότε, η ορατή σε αυτόν πλευρά θα είναι η μπροστά.*
  - ✓ *Εάν ο θεατής δηλώσει τις κορυφές του πολυγώνου με δεξιόστροφη φορά από τη δική του οπτική τότε, η ορατή σε αυτόν πλευρά θα είναι η πίσω.*



Γραφικά Υπολογιστών

Ρύθμιση όψεων πολυγώνων

- ✓ *Η σύμβαση στη δήλωση όψεων πολυγώνων μπορεί να μεταβληθεί από τον προγραμματιστή με την εντολή glFrontFace:*
  - ✓ *void glFrontFace(GLenum mode);*
  - ✓ *mode δέχεται τις συμβολικές σταθερές:*
  - ✓ *GL_CCW: Η ορατή στο θεατή όψη χαρακτηρίζεται ως μπροστινή όψη (front facing polygon) εάν οι κορυφές του πολυγώνου δηλωθούν κατά τη θετική φορα από την οπτική γωνία του θεατή (counterclockwise).*
  - ✓ *GL_CW: Η ορατή στο θεατή όψη χαρακτηρίζεται ως μπροστινή όψη εάν οι κορυφές του πολυγώνου δηλωθούν κατά την αρνητική φορα από την οπτική γωνία του θεατή (clockwise).*

Γραφικά Υπολογιστών

Τροποποίηση σχεδίασης πολυγώνων

- ✓ *Κάθε όψη ενός πολυγώνου (μπροστινή και πίσω) μπορεί να σχεδιαστεί με διαφορετικό τρόπο.*
- ✓ *Με την αρχική ρύθμιση της OpenGL, και οι μπροστινές και οι πίσω όψεις σχεδιάζονται συμπαγείς.*
- ✓ *Η ρύθμιση του τρόπου σχεδίασης κάθε όψης δηλώνεται με την εντολή glPolygonMode.*
  - ✓ *void glPolygonMode(GLenum face, GLenum mode);*

Γραφικά Υπολογιστών

Αλλαγή σχεδίασης όψεων πολυγώνων

- *void glPolygonMode(GLenum face, GLenum mode);*
- *face – η όψη της οποίας το σχεδιασμό τροποποιούμε:*
- *GL_FRONT: Η επιβαλλόμενη τροποποίηση αφορά τις μπροστινές όψεις*
- *GL_BACK: Η επιβαλλόμενη τροποποίηση αφορά τις πίσω όψεις*
- *GL_FRONT_AND_BACK: Η επιβαλλόμενη τροποποίηση αφορά και τις δύο όψεις.*
- *mode – τρόπος σχεδιασμού της επιλεγόμενης όψης:*
  - *GL_FILL: Η όψη σχεδιάζεται συμπαγής.*
  - *GL_LINE: Σχεδιάζεται μόνο το περίγραμμα της όψης.*
  - *GL_POINT: Σχεδιάζονται μόνο οι κορυφές της όψης.*
- *Παράδειγμα:*
  - *glPolygonMode(GL_FRONT,GL_LINE);*
  - *glPolygonMode(GL_BACK,GL_POINT);*
  - *Σχεδίαση των μπροστινών όψεων ως περιγράμματα*
  - *Σχεδίαση κορυφών των πίσω όψεων*
  - *glPolygonMode(GL_FRONT,GL_LINE);*
  - *Σχεδίαση των μπροστινών όψεων ως περιγράμματα*
  - *Σχεδίαση των πίσω όψεων κατά την προηγουμένως ισχύουσα ρύθμιση*

Γραφικά Υπολογιστών

Καταστολή όψεων πολυγώνων

- ✓ *Όταν σε μια σκηνή είναι ορατό μόνο ένα από τα δύο είδη όψεων πολυγώνων, συμφέρει να αγνοήσουνε το σχεδιασμό των μή ορατών όψεων.*
- ✓ *Η δυνατότητα καταστολής όψεων πολυγώνων αρχικά πρέπει να ενεργοποιηθεί με την εντολή glEnable*
  - ✓ *glEnable(GL_CULL_FACE);*
- ✓ *Οι όψεις που καταστέλλονται δηλώνονται με την εντολή glCullFace.*
  - ✓ *void glCullFace(GLenum mode);*
  - ✓ *void glCullFace(GLenum mode);*
  - ✓ *mode: δέχεται τις συμβολικές σταθερές*
  - ✓ *GL_FRONT: καταστολή μπροστινών όψεων*
  - ✓ *GL_BACK: καταστολή πίσω όψεων*
  - ✓ *GL_FRONT_AND_BACK: καταστολή και των δύο ειδών όψεων*
- ✓ *Παράδειγμα:*
  - ✓ *glCullFace(GL_BACK); Καταστολή πίσω όψεων*
  - ✓ *Ενεργοποίηση της καταστολής όψεων με την glEnable χωρίς χρήση της glCullFace αυτομάτως ενεργοποιεί την καταστολή των πίσω όψεων.*

Γραφικά Υπολογιστών

Ομάδες ιδιοτήτων (attribute groups)

- ✓ *Η χρήση πολλαπλών εντολών επισκόπησης μεταβλητών κατάστασης (glGetFloatv glGetDoublev κοκ) για την ανάκτηση ενός συνόλου τους είναι κουραστική για τον προγραμματιστή.*
- ✓ *Στην OpenGL οι ιδιότητες κατάστασης έχουν ομαδοποιηθεί σε ομάδες.*
- ✓ *Με τη χρήση μίας μόνο εντολή εκτελείται η αποθήκευση όλων των ιδιοτήτων μιας ομάδας σε μία θέση μνήμης για μελλοντική χρήση*
- ✓ *Οι ομάδες ιδιοτήτων αποθηκεύονται στη στοίβα ιδιοτήτων (attribute stack).*
- ✓ *Ορισμένες ομάδες ιδιοτήτων:*
  - ✓ *Ομάδα ιδιοτήτων σημείου:*
    - ✓ *Περιέχει παραμέτρους που καθορίζουν την εμφάνιση ενός σημείου (π.χ. Πάχος σημείου)*
  - ✓ *Ομάδα ιδιοτήτων γραμμών:*
    - ✓ *Περιέχει παραμέτρους που καθορίζουν την εμφάνιση γραμμών (π.χ. Πάχος γραμμής, διάστιξη γραμμής)*
  - ✓ *Ομάδα ιδιοτήτων πολυγώνων:*
    - ✓ *Εμπεριέχει όλες τις παραμέτρους που επηρεάζουν τη σχεδίαση πολυγώνων.*
  - ✓ *Χρώμα:*
    - ✓ *Το χρώμα εντάσσεται στη δική του "ομάδα ιδιοτήτων".*

Γραφικά Υπολογιστών

Αποθήκευση ομάδων ιδιοτήτων

- ✓ *Οι μεταβλητές μιας ομάδας αποθηκεύονται στη στοίβα ιδιοτήτων με την εντολή glPushAttrib:*
  - ✓ *void glPushAttrib(attributeGroup);*
  - ✓ *attributeGroup καθορίζει την ομάδα ιδιοτήτων*
  - ✓ *GL_POINT_BIT: ομάδα ιδιοτήτων σημείου*
  - ✓ *GL_LINE_BIT: ομάδα ιδιοτήτων γραμμής*
  - ✓ *GL_POLYGON_BIT: ομάδα ιδιοτήτων πολυγώνου*
  - ✓ *GL_CURRENT_BIT: "ομάδα ιδιοτήτων" χρώματος*
- ✓ *Παράδειγμα:*
  - ✓ *glPushAttrib(GL_CURRENT_BIT);*
  - ✓ *Αποθήκευση τρέχοντος χρώματος σχεδίασης στη στοίβα ιδιοτήτων.*
- ✓ *Οι τιμές ιδιοτήτων που αποθηκεύτηκαν στη στοίβα ιδιοτήτων ανακαλούνται με την εντολή glPopAttrib: void glPopAttrib( );*
- ✓ *Η glPopAttrib: ανακαλεί όλες τις τιμές των μεταβλητών κατάστασης που αποθηκεύτηκαν στο παρελθόν στη στοίβα ιδιοτήτων με εντολές glPushAttrib και ανακαλεί μόνο τις τιμές των μεταβλητών κατάστασης που έχουν αποθηκευτεί στο παρελθόν στη στοίβα ιδιοτήτων.*

Γραφικά Υπολογιστών

Λίστες απεικόνισης (display lists)

- ✓ *Η περιγραφή ενός σύνθετου γεωμετρικού σήματος είναι βολικό να περικλείεται σε μια αυτόνομη ενότητα κώδικα, τη λίστα απεικόνισης (display list).*
- ✓ *Η λίστα απεικόνισης εκτελείται κάθε φορά που θέλουμε να σχεδιάσουμε το σύνθετο σχήμα. Δίνει τη δυνατότητα επεναχρησιμοποίησης κώδικα.*
- ✓ *Σε κάθε λίστα απεικόνισης καταχωρείται ένας ακέραιος αναγνωριστικός αριθμός (list identifier)*
- ✓ *Δημιουργούμε αναγνωριστικούς αριθμούς με την εντολή glGenLists.*
  - ✓ *GLuint glGenLists(GLint range);*
  - ✓ *range: το πλήθος των αναγνωριστικών ακεραίων που θα δημιουργήσουμε*
- ✓ *Παράδειγμα:*
  - ✓ *ListID = glGenLists(3);*
  - ✓ *Δημιουργία τριών αναγνωριστικών με τιμές listID, listID+1, listID+2*

Γραφικά Υπολογιστών

Δημιουργία λίστας απεικόνισης

- ✓ *Μια λίστα απεικόνισης ορίζεται μεταξύ δύο εντολών: των glNewList και glEndList.*
  - ✓ *void glNewList(GLuint listID, GLenum listMode);*
  - ✓ *listID: ο αναγνωριστικός που αποδίδουμε στη λίστα απεικόνισης*
  - ✓ *listMode: δέχεται τις σταθερές*
  - ✓ *GL_COMPILE: για απλή δήλωση του κώδικα σχεδιασμού*
  - ✓ *GL_COMPILE_AND_EXECUTE: για δήλωση και εκτέλεση του κώδικα σχεδιασμού*
  - ✓ *glNewList(...);*
  - ✓ *Εντολές δήλωσης σχήματος*
  - ✓ *glEndList();*
  - ✓ *Μεταξύ των εντολών glNewList και glEndList ορίζουμε το σύνθετο γεωμετρικό σχήμα χρησιμοποιώντας τις εντολές σχεδιάσης σχημάτων που προαναφέρθηκαν.*

Γραφικά Υπολογιστών

Εκτέλεση/Διαγραφή λίστας απεικόνισης

- ✓ *Μια λίστα απεικόνισης εκτελείται δίνοντας το αναγνωριστικό της στην εντολή glCallList:*
  - ✓ *void glCallList(GLuint listID);*
- ✓ *Οι λίστες απεικόνισης διαγράφονται με την εντολή glDeleteLists. (Οι αναγνωριστικοί αριθμοί τους αποδεσμεύονται.)*
  - ✓ *glDeleteLists(startId, nLists);*
  - ✓ *startID: ο αναγνωριστικός της πρώτης λίστας που διαγράφεται*
  - ✓ *nLists: το πλήθος των λιστών που διαγράφονται.*

- ✓ *Παράδειγμα:*
  - ✓ *glDeleteLists(someListID,3);*
  - ✓ *Διαγράφουμε τις λίστες απεικόνισης με αναγνωριστικούς αριθμούς someListID, someListId+1 και someListId+2.*

Γραφικά Υπολογιστών

Λίστες απεικόνισης και μεταβλητές κατάστασης

- ✓ *Εάν μεταβάλλουμε μέσα σε μια display list την τιμή μιας μεταβλητής κατάστασης (όπως π.χ. του τρέχοντος χρώματος σχεδίασης), η μεταβολή αυτή θα παραμείνει ενεργή και μετά το πέρας εκτέλεσης της λίστας.*
- ✓ *Είναι χρήσιμοο προγραμματιστής να αποθηκεύσει τις τιμές των ιδιοτήτων που θα μεταβληθούν στη στοίβα ιδιοτήτων, πριν από την κλήση της λίστας, ούτως ώστε να τις επαναφέρει μετά το πέρας της εκτέλεσης της λίστας.*

Μητρώα κορυφών - Μητρώα χρωμάτων

- ✓ *Ο σχεδιάσμός μεγάλου πλήθους σχημάτων αποκλειστικά με τη χρήση της δομής glBegin/glEnd απαιτεί έναν υπερβολικά μεγάλο αριθμό εντολών*
- ✓ *Για να διευκολύνει τη σχεδίαση πολλαπλών σχημάτων, η OpenGL παρέχει την τεχνική των μητρώων κορυφών (vertex arrays).*
- ✓ *Με τα μητρώα σημείων σχεδιάζουμε ένα μεγάλο αριθμό σχημάτων δηλώνοντας απλώς ένα μητρώο κορυφών και τον είδος των σχηματων που αυτές ορίζουν.*
- ✓ *Η ίδια ιδέα επεκτείνεται και στην απόδοση χρωματικών τιμών σε μεγάλο πλήθος κορυφών. Στην περίπτωση αυτή ορίζουμε τα μητρώα χρωμάτων (color arrays).*

Γραφικά Υπολογιστών

Ενεργοποίηση χρήσης μητρώου κορυφών/χρωμάτων

- ✓ *Αρχικά, απαιτείται η ενεργοποίηση της χρήσης μητρώων σημείων ή/και χρωμάτων με την εντολή glEnableClientState:*
  - ✓ *void glEnableClientState(GLenum array);*
  - ✓ *array: δηλώνει την κατηγορία των δεδομένων που ομαδοποιούνται στο μητρώο*
  - ✓ *GL_VERTEX_ARRAY: ομαδοποίηση συντεταγμένων πολλαπλών κορυφών*
  - ✓ *GL_COLOR_ARRAY: ομαδοποίηση χρωματικών τιμών πολλαπλών κορυφών.*
  - ✓ *GL_TEXTURE_ARRAY: αναλύεται στο Κεφάλαιο "Απόδοση υφής"*

Γραφικά Υπολογιστών

Ομαδοποίηση κορυφών

- ✓ *Βήμα 1ο: Ενεργοποιούμε τη χρήση μητρώων κορυφών με την glEnableClientState.*
  - ✓ *glEnableClientState(GL_VERTEX_ARRAY);*
- ✓ *Βήμα 2ο: Καταχωρούμε το μητρώο κορυφών με την εντολή glVertexPointer.*
  - ✓ *void glVertexPointer(GLint size, GLenum type, GLsizei stride, const Glvoid *vertexPointer);*
  - ✓ *vertexPointer: δείκτης στο μητρώο κορυφών*
  - ✓ *size: το πλήθος των τιμών που προσδιορίζουν τις συντεταγμένες καθεμιάς κορυφής (για διδιάστατες κορυφές: 2 και για τρισδιάστατες κορυφές: 3)*
  - ✓ *void glVertexPointer(GLint size, GLenum type, GLsizei stride, const GLvoid *vertexPointer);*
  - ✓ *type: ο πρωτογενής τύπος δεδομένων με τον οποίο αποθηκεύονται οι συντεταγμένων στο μητρώο vertexPointer.*
  - ✓ *Αριθμητικές σταθερές: GL_INT, GL_SHORT, GL_FLOAT, GL_DOUBLE*
  - ✓ *stride: η απόσταση μεταξύ των συντεταγμένων διαδοχικών σημείων στο μητρώο. Χρησιμοποιείται μόνο σε όταν αποθηκεύονται τιμές πολλών ιδιοτήτων στο ίδιο μητρώο (πχ διαδοχική αποθήκευση συντεταγμένων και χρώμάτων σε ένα μητρώο) Για μητρώα που περιέχουν μόνο συντεταγμένες κορυφών η τιμή 0).*

Γραφικά Υπολογιστών

Ομαδοποίηση κορυφών

- ✓ *Ορίζουμε τη διαδοχή με την οποία χρησιμοποιούνται τα σημεία του μητρώου κορυφών. Η διαδοχή ορίζεται σε ένα μητρώο δεικτών.*
- ✓ *Το μητρώο δεικτών ορίζει τη σειρά χρήσης των σημείων του μητρώου κορυφών. Η κορυφή που δηλώνεται στη θέση i1 του μητρώου κορυφών χρησιμοποιείται πρώτη, η κορυφή που δηλώνεται στη θέση i2 του μητρώου κορυφών χρησιμοποιείται δεύτερη κ.ο.κ.*
- ✓ *Βήμα 3ο: Εκτελούμε τη σχεδίαση των σχημάτων με την εντολή glDrawElements:*
  - ✓ *void glDrawElements(GLenum mode, GLsizei count, GLenum type, Glvoid *indices);*
  - ✓ *indices: δείκτης στο μητρώο δεικτών*
  - ✓ *mode: καθορίζει τι σχήματα ορίζουν οι κορυφές. (GL_LINES, GL_TRIANGLES, GL_QUADS κ.λ.π.).*
  - ✓ *count: το πλήθος των κορυφών που περιέχονται στο μητρώο κορυφών.*
  - ✓ *type: ο πρωτογενής τύπος δεδομένων των τιμών στο μητρώο δεικτών.*
  - ✓ *Δεκτές αριθμητικές σταθερές: GL_UNSIGNED_BYTE, GL_UNSIGNED_SHORT, GL_UNSIGNED_INT.*

Γραφικά Υπολογιστών

Ομαδοποίηση χρωματικών τιμών

- ✓ *Η ομαδοποίηση χρωματικών τιμών εκτελείται με ακριβώς τον ίδιο τρόπο.*
  - ✓ *glEnableClientState(GL_COLOR_ARRAY);*
- ✓ *Το μητρώο χρωματικών τιμών καταχωρείται με την εντολή glColorPointer.*
  - ✓ *void glColorPointer(GLint size, GLenum type, GLsizei stride, const Glvoid *vertexPointer);*
  - ✓ *size: το πλήθος των χρωματικών συνιστωσών (για το μοντέλο RGB: 3 και για το μοντέλο RGBA: 4)*
- ✓ *Ορίζουμε ένα μητρώο δεικτών και δίνουμε την εντολή σχεδίασης με την glDrawElements όπως προηγουμένως.*

Γραφικά Υπολογιστών

Βιβλιογραφία

- ✓ *https://www.opengl.org/.*
- ✓ *https://developer.nvidia.com/opengl.*
- ✓ *http://www.medialab.ntua.gr/.*
- ✓ *"Products: Software: OpenGL: Licensing and Logos". SGI.*
- ✓ *"OpenGL 3.0 Released, Developers Furious – Slashdot". Tech.slashdot.org.*
- ✓ *"OpenGL 4.5 released—with one of Direct3D's best features". Ars Technica.*