Αλληλεπίδραση

- Project sketchpad: πρώτο αλληλεπιδραστικό πρόγραμμα γραφικών
- Αλληλεπίδραση βασικό συστατικό προγραμμάτων γραφικών
- Η OpenGL δεν υποστηρίζει άμεσα αλληλεπίδραση (συναρτήσεις διαχείρισης παραθύρων και συσκευών εισόδου) για λόγους μεταφερσιμότητας.
- Υποστήριξη από τη GLUT

Συσκευές εισόδου

- Φυσικές συσκευές εισόδου
- Λογικές συσκευές εισόδου
  - Τι δυνατότητες προσφέρουν στην εφαρμογή.
  - Δεν χαρακτηρίζονται από τις φυσικές τους ιδιότητες αλλά από τον τρόπο που επικοινωνούν με το πρόγραμμα μέσω συναρτήσεων

Συσκευές εισόδου

- printf : η φυσική συσκευή εξόδου δεν ενδιαφέρει.
- Μια φυσική συσκευή ως δύο διαφορετικές λογικές συσκευές
  - Ποντίκι: θέση ή κωδικός menu

Φυσικές συσκευές εισόδου

- Συσκευές κατάδειξης (pointing devices)
  - Ποντίκι
  - trackball
  - data tablet
  - lightpen
  - joystick
  - spaceball
- Πληκτρολόγια
- Data gloves, tracking devices.

Κατηγορίες λογικών συσκευών

- Τι μετρήσεις επιστρέφει η συσκευή στο πρόγραμμα
- Πότε επιστρέφει τις μετρήσεις

- Συμβολοσειράς (string)
  - Πληκτρολόγιο
- Εντοπιστής (locator)
  - Θέση
  - Ποντίκι, trackball

Κατηγορίες λογικών συσκευών

- Λήψης (pick)
  - Επιστρέφει τον χαρ. αριθμό ενός αντικειμένου
  - Διαδικασία επιλογής (selection) στην OpenGL
- Επιλογή (choice)
  - Μια από ένα σύνολο διακριτών επιλογών
  - Υλοποίηση με widgets (menus, κουμπιά)

Κατηγορίες λογικών συσκευών

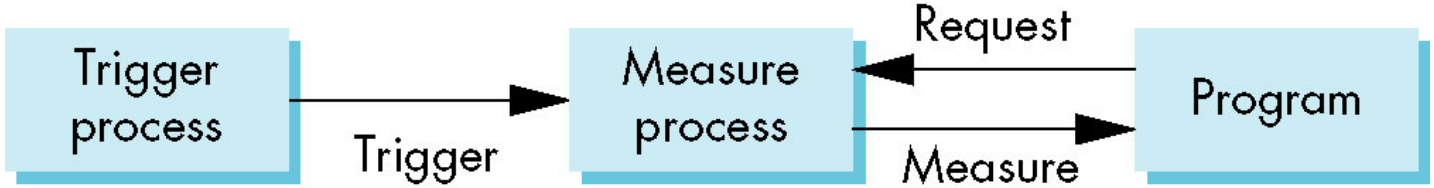
- Αναλογικός επιλογέας (dial)
  - Παρέχει αναλογική είσοδο
  - Υλοποιείται με widgets (slidebars)
- Stroke
  - Επιστρέφει μια σειρά θέσεων

Μέτρηση & έναυση (measure & trigger)

- Μέτρηση: τι επιστρέφει η συσκευή στο πρόγραμμα
- Έναυση: είσοδος στη συσκευή που στέλνει σήμα στο πρόγραμμα
  - Πληκτρολόγιο: συμβολοσειρά- enter
  - Ποντίκι: θέση – πάτημα πλήκτρου
- Η μέτρηση μπορεί να περιέχει και ενδείξεις κατάστασης.

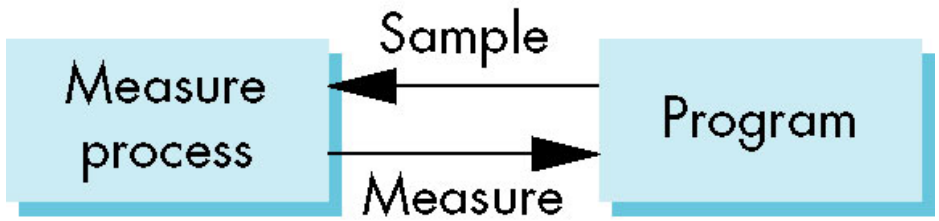
Τρόποι εισόδου

- Μετά από αίτηση (request mode)
  - Η εφαρμογή ζητάει μέτρηση
  - Η μέτρηση επιστρέφεται στο πρόγραμμα μόνο όταν διεγερθεί η συσκευή εισόδου
  - Scanf & enter



Τρόποι εισόδου

- Με δειγματοληψία (sample mode)
  - Η μέτρηση επιστρέφεται στο πρόγραμμα μόλις γίνει κλήση της αντίστοιχης συνάρτησης
  - Δεν χρειάζεται έναυσμα

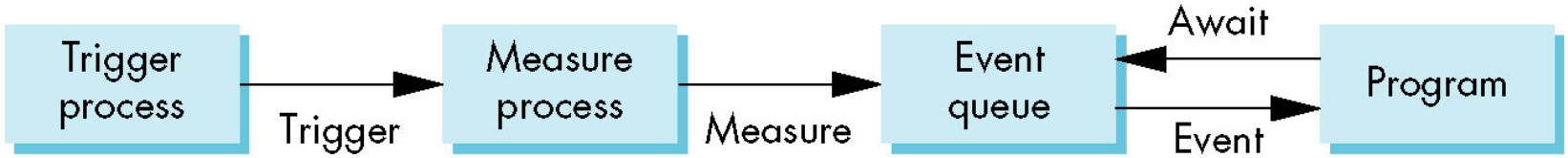


Τρόποι εισόδου

- Στους δύο αυτούς τρόπους το πρόγραμμα δέχεται είσοδο μόνο από τις συσκευές που θα επιλέξει ο προγραμματιστής.
  - Χρήσιμες για προγράμματα που καθοδηγούν τον χρήστη (γραμμικά) αλλά όχι για αυτά που ο χρήστης ελέγχει τη ροή.

Τρόποι εισόδου

- Οδηγούμενος από γεγονότα (event mode)
  - Ιδανικός για την περίπτωση πολλών συσκ. εισόδου.
  - Κάθε έναυση σε μια συσκευή δημιουργεί ένα γεγονός (μέτρηση+χαρακτηριστικό συσκευής).
  - Τοποθέτηση στην ουρά γεγονότων
  - Το πρόγραμμα αποφασίζει τι θα κάνει με τα γεγονότα
  - Συσχέτιση callback με κάθε τύπο γεγονότος



Callback functions

- Συσχέτιση (registration) συναρτήσεων με γεγονότα που αφορούν την εφαρμογή μας.
  - glutMouseFunc(mouse_callback_func)
  - Η callback λαμβάνει τα ορίσματά της από το σύστημα.
- Αρχικοποίηση κύκλου γεγονότων με glutMainLoop
  - Το πρόγραμμα περιμένει για γεγονότα και μόλις συμβεί κάτι ενδιαφέρον εκτελείται η σχετική callback function

Γεγονότα που σχετίζονται με το ποντίκι

- Γεγονός ποντικιού: πάτημα/άφημα πλήκτρου ποντικού
  - ***glutMouseFunc****(void (*func)(int button, int state, int x, int y));*
- Κίνηση: κίνηση με πατημένο πλήκτρο
  - ***glutMotionFunc****(void (*func)(int x, int y));*
- Παθητική κίνηση: κίνηση χωρίς πατημένο πλήκτρο
  - ***glutPassiveMotionFunc****(void (*func)(int x, int y));*

```
void mouse_callback_func(int button, int state, int x,
    int y)
{
if(button==GLUT_LEFT_BUTTON &&
    state==GLUT_DOWN)
exit();
}
glutMouseFunc(mouse_callback_func)
```

- Σε άλλα γεγονότα δεν συμβαίνει τίποτα

Άλλες callback functions

- Γεγονότα πληκτρολογίου
  - glutKeyboardFunc(*void (*func)(unsigned char key, int x, int y)*)
- glutDisplayFunc()
  - Η σχετική callback καλείται όταν χρειάζεται να επιδειχθεί ή ξαναεπιδειχθεί το παράθυρο
  - Απαραίτητη στα προγράμματα
- glutPostRedisplay()

Άλλες callback functions

- glutIdleFunc()
  - Η σχετική callback καλείται όταν δεν υπάρχει γεγονός
- Μπορούμε να αλλάξουμε/απενεργοποιήσουμε τις callback που συσχετίζονται με ένα γεγονός

Menus

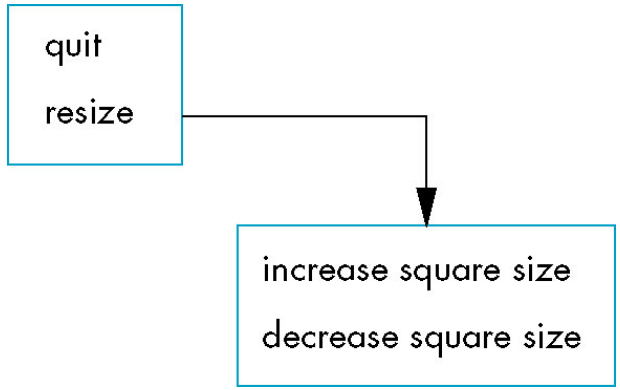
- Δημιουργία pop-up menus
  - Ορισμός επιλογών του menu
  - Συσχέτιση του menu με πλήκτρο του ποντικιού
  - Δημιουργία callback για προσδιορισμό ενεργειών

Menus

```
glutCreateMenu(demo_menu);
    glutAddMenuEntry("quit",1);
    glutAddMenuEntry("lala",2);
  glutAttachMenu(GLUT_RIGHT_BUTTON
  );
void demo_menu(int id)
{ if(id==1) exit(0);}
```

Menus

- Δημιουργία υπο-menu




```
Sub_menu=glutCreateMenu(size_menu);
glutAddMenuEntry(“increase",2);
 glutAddMenuEntry(“decrease",3);
glutCreateMenu(top_menu);
 glutAddMenuEntry(“quit",1);
 glutAddSubMenu (“Resize",sub_menu);
glutAttachMenu(GLUT_RIGHT_BUTTON);
```

- Πρόγραμμα square.c
  - Ζωγράφισμα τετραγώνων κάθε φορά που το ποντίκι κινείται με πατημένο το αριστερό πλήκτρο.
  - Σχεδίαση μέσα στην callback κίνησης ***glutMotionFunc(motion)***
  - Χειρισμός της αλλαγής διάστασης παραθύρου
    - glutReshapeFunc(*void (*func)(int x, int y));*
  - *Χειρισμός των συντεταγμένων που γυρνάει το ποντίκι.*

Πολλαπλά παράθυρα

- id=glutCreateWindow(“”);
- glutSetWindow(id);
- Κλήση της glutInitDisplayMode για αλλαγή ιδιοτήτων
- Κάθε παράθυρο μπορεί να έχει δικές του callback functions

glutInitDisplayMode();

id=glutCreateWindow(“”);

Callback registration

Display lists

- Ομαδοποίηση/αποθήκευση εντολών για μελλοντική χρήση/επανάχρηση

glNewList(1, GL_COMPILE)

…..

glEndList()

….

glCallList(1)

- Σε περίπτωση μοντέλου client-server η λίστα στέλνεται για αποθήκευση στον server-μείωση του όγκου δεδομένων στο δίκτυο.
- Βελτιστοποίηση του κώδικα.

Display lists

- Σχεδιασμός / αποθήκευση γραμματοσειράς

```
base=glGenLists(256)
for(i=0; i<256; i++){
glNewList(base+i, GL_COMPILE);
myfont(i);
glEndList();}
```

Display lists

- Σχεδιασμός / αποθήκευση γραμματοσειράς

```
glListBase(base);
char* text_string;
glCallLists((GLint) strlen(text_string),
   GL_BYTE; text_string);
```

Display lists

- GL_COMPILE_AND_EXECUTE

- Κατά την κλήση της λίστας τα αντικείμενα της σχεδιάζονται με βάση την τρέχουσα κατάσταση του συστήματος
  - Σχεδίαση του ίδιου αντικειμένου σε διαφορετικές θέσεις
- Αλλαγή κατάστασης μέσα στη λίστα

Σωροί (stacks)

- Αποθήκευση μεταβλητών κατάστασης & πινάκων model-view & projection.
- Ανάκληση όταν απαιτηθεί
- glPushAttrib(GL_ALL_ATTRIB_BITS)
- glPopAttrib()
- glPushMatrix()
- glPopMatrix()
  - Επιδρούν στον ενεργό πίνακα

GLUT fonts

glRasterPos2i(rx,ry);

glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_10, k);

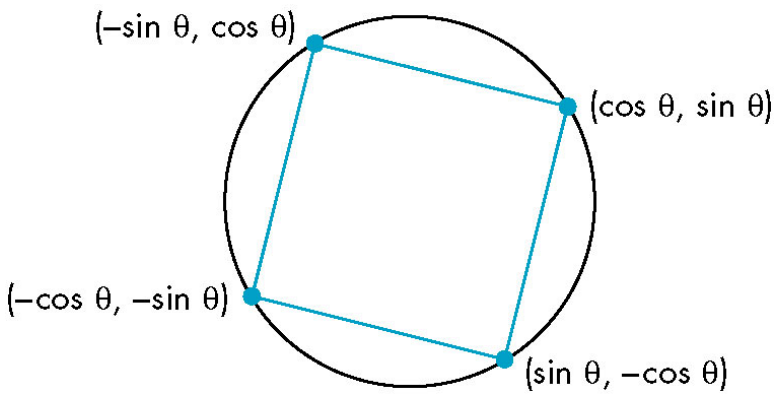
- Η glutBitmapCharacter μετακινεί αυτόματα το raster position στην θέση για το επόμενο γράμμα.

glutBitmapWidth(GLUT_BITMAP_TIMES_ROMAN_10, k);

glutStrokeCharacter(GLUT_STROKE_MONO_roman, k);

Γραφικά με κίνηση

- Επίδειξη περιστρεφόμενου τετραγώνου
- (cosθ, sinθ), (-cosθ, -sinθ), (- sinθ, cosθ), (sinθ, -cosθ): σχηματίζουν τετράγωνο πάνω σε μοναδιαίο κύκλο.



```
void display(){
glClear(GL_COLOR_BUFFER_BIT);
glBegin(GL_POLYGON);
thetar=theta/((2*3.141)/360.0);
glVertex2f(cos(thetar), sin(thetar));
…
glEnd();}
```

- Αλλαγή θ: περιστροφή του τετραγώνου
- Περιστροφή τετραγώνου όταν δεν συμβαίνει κάτι

```
glutIdleFunc(idle);
void idle(){
theta+=2;
if(theta>=360.0) theta-=360.0;
glutPostRedisplay();}

glutMouseFunc(mouse);

void mouse(int button, int state, int x, int y){
if(button==GLUT_LEFT_BUTTON&&
    state==GLUT_DOWN)
glutIdleFunc(idle);
if(button==GLUT_MIDLE_BUTTON&&
    state==GLUT_DOWN)
glutIdleFunc(NULL);}
```

Double buffering

- Ο frame buffer απεικονίζεται στην οθόνη ανά τακτά χρονικά διαστήματα (refresh rate).
- Προβλήματα όταν:
  - Αλλάζουμε τον frame buffer καθώς γίνεται ξαναζωγράφισμα (refresh) της οθόνης
  - Η αλλαγή διαρκεί πιο πολύ από ένα κύκλο
- Χρήση δύο buffers: front/back buffer.
- Επιδεικνύεται ο front buffer, ζωγραφίζουμε στον back buffer

Double Buffering

- Μόλις είμαστε έτοιμοι αλλάζουμε τους buffers
- glutSwapBuffers(): οι buffers εναλλάσσονται στον επόμενο κύκλο refresh.
- glutInitDisplayMode(GLUT_RGB|GLUT_DOUBLE)
- Το double buffering δεν επιταχύνει τη διαδικασία ζωγραφίσματος.
- single_double.c

Double Buffering

- Επιλογή του buffer όπου θα γίνει το ζωγράφισμα

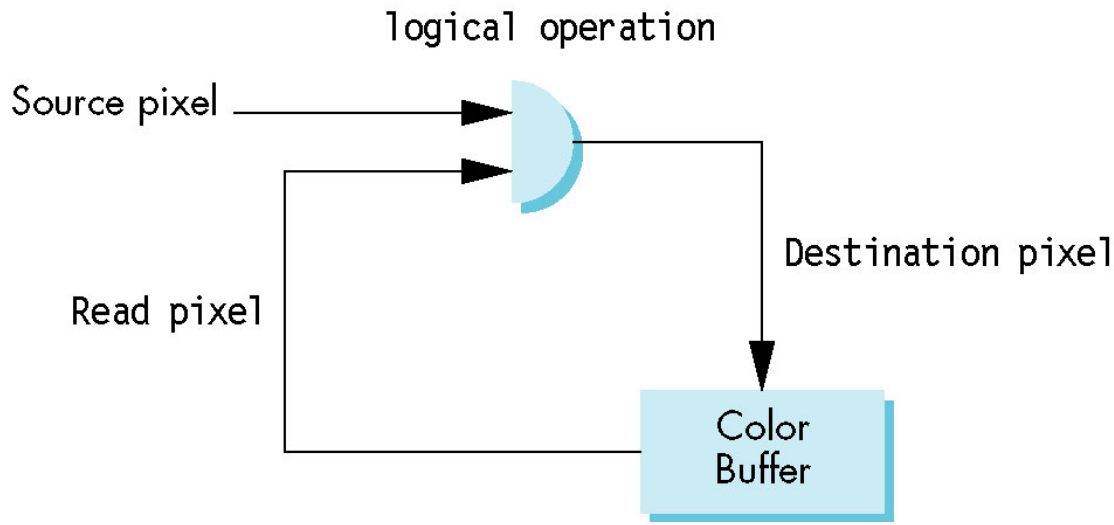
glDrawBuffer(GL_BACK) (default)

glDrawBuffer(GL_FRONT_AND_BACK)

Logic Operations

- Εξ ορισμού οι εντολές ζωγραφίσματος αντικαθιστούν τα αντίστοιχα pixels του framebuffer.
- Αντί αυτού μπορούμε να έχουμε γράψιμο που συνδυάζει με λογικές πράξεις την νέα (source) με την τρέχουσα (destination) τιμή ενός pixel για να παραχθεί η νέα τιμή

Logic Operations



glEnable(GL_COLOR_LOGIC_OP);

glLogicOp(GL_XOR);

Logic Operations

- Οι λογικές πράξεις γίνονται σε κάθε bit χωριστά).
- Writing modes: 16
- XOR mode: d'=d XOR s
- d' XOR s= (d XOR s) XOR s=d
- Για να σβήσουμε κάτι που έχουμε ζωγραφίσει με XOR mode απλά το ξαναζωγραφίζουμε.

Rubberbanding

Globals: xm,ym,xmm,ymm

```
void mouse(int btn, int state, int x, inty){
if(btn==GLUT_LEFT_BUTTON &&
   state==GLUT_DOWN){
xm=x; ym=(500-y);
glLogicOp(GL_XOR);
first=0;}
```

Rubberbanding

```
If(btn==GLUT_LEFT_BUTTON &&
   state==GLUT_UP){
glRectf(xm, ym, xmm, ymm); //σβησιμο
glLogicOp(GL_COPY);
xmm=x; ymm=500-y;
glRectf(xm, ym, xmm, ymm);} //τελικό
                            //γράψιμο
}
```

Rubberbanding

```
void move(int x, int y){
if(first==1){
glRectf(xm, ym, xmm, ymm);}//σβησιμο
xmm=x; ymm=500-y;
glRectf(xm, ym, xmm, ymm); //γράψιμο
first=1;}
```

Logic Operations

- Τα χρώματα που παίρνουμε με XOR mode εξαρτώνται από το τρέχων χρώμα του buffer.
- (11111111 11111111 11111111) XOR (00000000 00000000 11111111)= (11111111 11111111 00000000)