Input and Interaction

Fall 2025
9/25/2025
Kyoung Shin Park
Computer Engineering
Dankook University

Overview

- Introduce the basic input devices
  - Physical input devices
    - Mouse, Keyboard, Trackball
  - Logical input devices
    - String, Locator, Pick, Choice, Valuators, Stroke device
- Input modes
  - Request mode
  - Sample mode
  - Event mode
- Devices & Event-driven programming
  - mouse, keyboard,..

Interaction

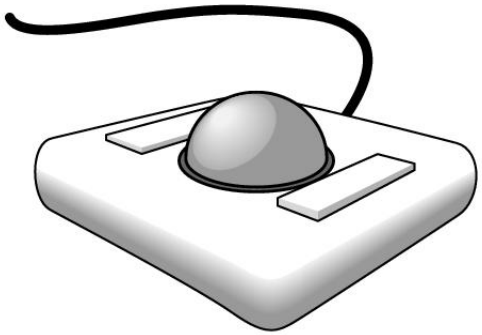
- One of the major advances in computer technology is that users can interact using computer screens.
- Interaction
  - The user takes action through an interactive device such as a mouse.
  - The computer detects user input.
  - The program changes it state in response to this input.
  - The program displays this new status.
  - The users sees the changed display.
  - The processes in which the user reacts to this change are repeated.

Graphical Input

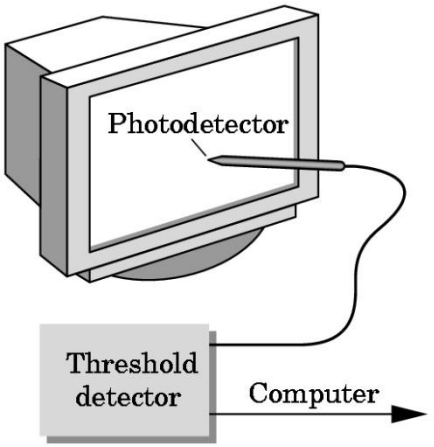
- Input devices can be described either by
  - Physical properties
    - Mouse, Keyboard, Trackball
  - Logical properties
    - Characterized by upper interface with application program, not by physical characteristics
- Input modes
  - The way an input device provides an input to an application program can be described as a **measurement** process and device **trigger**.
    - Request mode
    - Sample mode
    - Event mode

Physical Input Devices

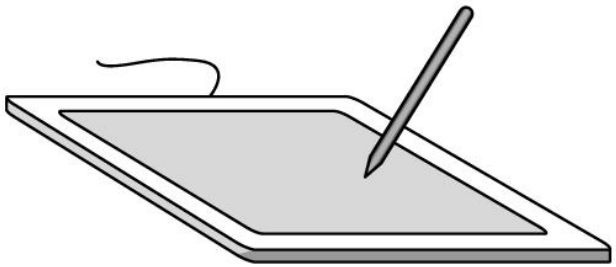
mouse



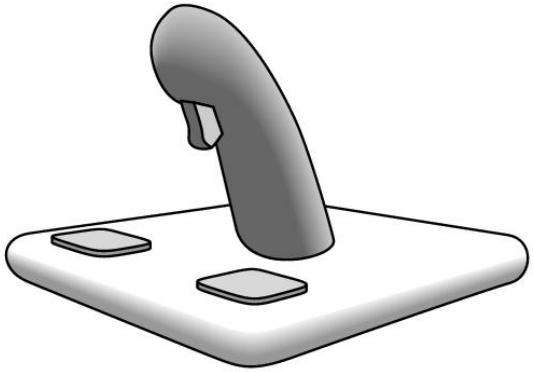
trackball



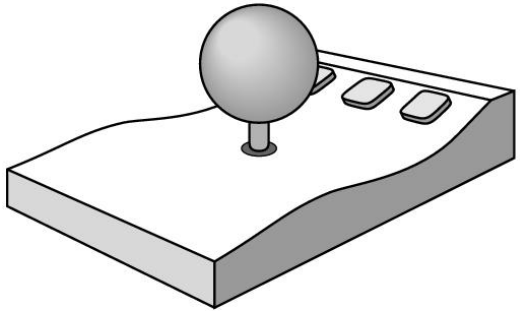
light pen



data tablet



joy stick



space ball

Physical Input Devices

- Physical input devices
  - Pointing devices
    - Allows the user to point to a location on the screen
    - In most cases, the user has more than one button to send a signal or interrupt to the computer.
    - Mouse, trackball, tablet, lightpen, joystick, spaceball
  - Keyboard devices
    - A device that returns a character code to a program
    - Keyboard

Relative Positioning Device

- Devices such as the data tablet return a position directly to the operating system
- Devices such as the mouse, trackball, and joy stick return incremental inputs (or velocities) to the operating system
  - Must integrate these inputs to obtain an absolute position
    - Rotation of cylinders in mouse
    - Roll of trackball
    - Difficult to obtain absolute position
    - Can get variable sensitivity

Logical Input Devices

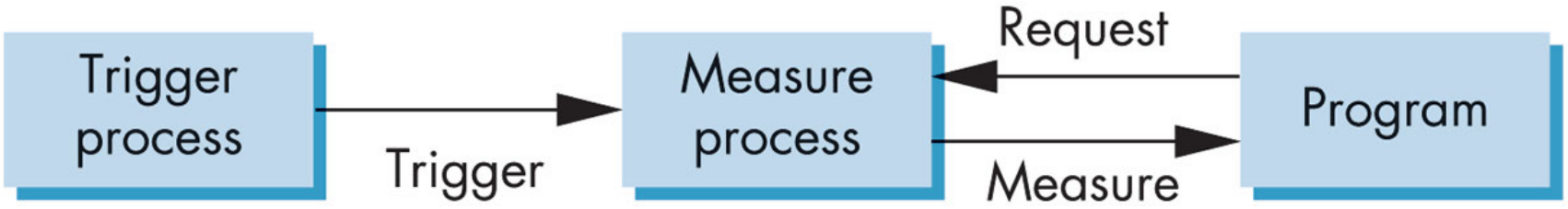
- String device - keyboard
  - Provide ASCII strings of characters to the program
- Locator device – mouse, trackball
  - Provide real world coordinate position to the program
- Pick device – mouse button, gun
  - Return the object's identifier(ID) to the program
- Choice device – widgets, function keys, mouse button
  - Let the user choose one of the options (menu)
- Valuators – slide bars, joystick, dial
  - Provide analog input (range of value) to the program
- Stroke – mouse drag
  - Return array of positions

Input Modes

- Input devices contain a ***trigger*** which can be used to send a signal to the operating system
  - Button on mouse
  - Pressing or releasing a key
- When triggered, input devices return information (their ***measure***) to the system
  - Mouse returns position information
  - Keyboard returns ASCII code

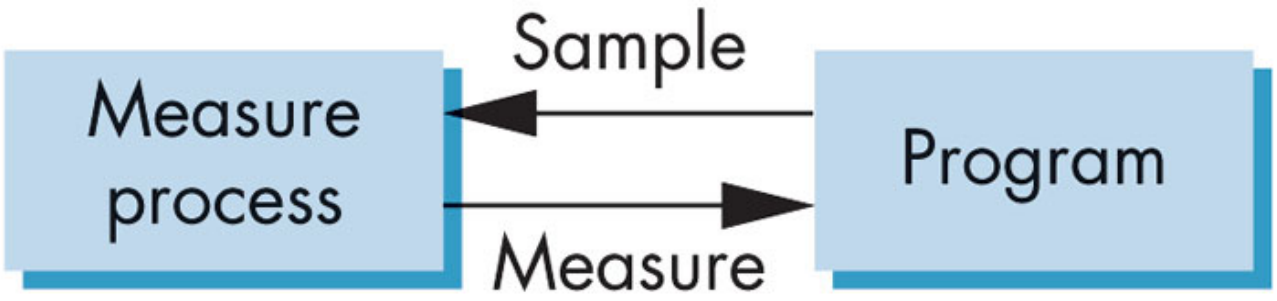
Request Mode

- In request mode, input measurement are not returned to the program until the user triggers the device.
- Standard for typical non-GUI program requiring character input
  - For example, when the C program's scanf function is used, the program stops while waiting for the terminal to type a character. Then, you can type and edit until you hit the enter-key(trigger).



Sample Mode

- Sample mode provides immediate input measures. As soon as the program encounters a function call, the measurement is returned. Therefore, no trigger is required.
- Example: getc function in C program



Event Mode

- Most systems have more than one input device, each of which can be triggered at an arbitrary time by a user.
- Each trigger generates an ***event*** whose measure is put in an ***event queue*** which can be examined by the user program.
- Use the callback function for a specific event.

GetAxis Unity Input vs InputSystem

- Axis - Unity Legacy Input Class
  - float h = Input.GetAxis("Horizontal");
  - float v = Input.GetAxis("Vertical");
  - float mouseX = Input.GetAxis("Mouse X");
  - float mouseY = Input.GetAxis("Mouse Y");
- Axis - Unity New Input System (on Input Action Asset)
  - // Actions "Move" (2D Vector) WASD, Arrow-Key, Gamepad stick
  - Vector2 move = actions.Move.ReadValue<Vector2>();
  - // Actions "Look" (2D Vector) Mouse Delta, Gamepad right stick
  - Vector2 look = actions.Look.ReadValue<Vector2>();

Unity Input vs InputSystem

- Button - Unity Legacy Input Class
  - if (Input.GetButtonDown("Fire1")) { ... }
  - if (Input.GetMouseButtonDown(0)) { ... }
- Button - Unity New Input System (on Input Action Asset)
  - if (actions.Fire.WasPressedThisFrame()) { ... }
  - if (Mouse.current.leftButton.wasPressedThisFrame) { ... }
  - if (Gamepad.current.buttonSouth.wasPressedThisFrame) { ... }

Unity Input vs InputSystem

- Key - Unity Legacy Input Class
  - if (Input.GetKey(KeyCode.UpArrow)) { ... }
  - if (Input.GetKeyDown(KeyCode.Space)) { ... }
- Key - Unity New Input System (on Input Action Asset)
  - if (Keyboard.current.upArrowKey.isPressed) { ... }
  - if (Keyboard.current.spaceKey.wasPressedThisFrame) { ... }

Unity Input vs InputSystem

- Mouse - Unity Legacy Input Class
  - Vector3 mousePos = Input.mousePosition;
- Mouse - Unity New Input System (on Input Action Asset)
  - Vector2 mousePos = Mouse.current.position.ReadValue();

Unity Input vs InputSystem

- Touch - Unity Legacy Input Class

```
if (Input.touchCount > 0)
{
    Touch t = Input.GetTouch(0);
    Vector2 delta = t.deltaPosition;
}
```

- Mouse - Unity New Input System (on Input Action Asset)

```
if (Touchscreen.current != null &&
Touchscreen.current.touches.Count > 0)
{
    var t = Touchscreen.current.touches[0];
    Vector2 delta = t.delta.ReadValue();
}
```

Keyboard Event Callback

- Call this function from the **Update()** function, since the state gets reset each frame.

```
using UnityEngine.InputSystem; // New Input System (Unity6)
public class KeyboardExample : MonoBehaviour {
    void Update()  {
        // The value is in the range -1 to 1 (y=Vertical & x=Horizontal)
        Vector2 move =
    InputSystem.actions["Move"].ReadValue<Vector2>();
        float translate = move.y * speed;
        float rotate = move.x * rotSpeed;
        // ESC-key exits the program
        if (Keyboard.current.escapeKey.wasPressedThisFrame)) {
              Application.Quit();
        }
    }
}
```

Mouse Functions

UnityEngine.Input	UnityEngine.InputSystem
Input.GetAxis("Mouse X")	Mouse.current.delta.ReadValue().x
Input.GetAxis("Mouse Y")	Mouse.current.delta.ReadValue().y
Input.GetMouseButton(int button)	Mouse.current.leftButton.isPressed Mouse.current.rightButton.isPressed Mouse.current.middleButton.isPressed
Input.GetMouseButtonDown(int button)	Mouse.current.leftButton.wasPressedThisFrame Mouse.current.rightButton.wasPressedThisFrame Mouse.current.middleButton.wasPressedThisFrame
Input.GetMouseButtonUp(int button)	Mouse.current.leftButton.wasReleasedThisFrame Mouse.current.rightButton.wasReleasedThisFrame Mouse.current.middleButton.wasReleasedThisFrame
Input.mousePosition	Mouse.current.position.ReadValue()
Input.mouseScrollDelta	Mouse.current.scroll.ReadValue() (Vector2)

Mouse Event Callback

```
public class MouseExample : MonoBehaviour {
    void Update()    {
        // 마우스 이동 (delta)
        Vector2 mouseDelta = Mouse.current.delta.ReadValue();
        Debug.Log($"Mouse Delta: {mouseDelta}");
        // 마우스 포지션
        Vector2 mousePos = Mouse.current.position.ReadValue();
        Debug.Log($"Mouse Position: {mousePos}");
        // 마우스 버튼 입력
        if (Mouse.current.leftButton.wasPressedThisFrame)
            Debug.Log("Left Mouse Button Down");
        // 스크롤 입력 (보통 y축 값 사용: up=+120, down=-120 → Unity에서
-1~0~1로 정규화 가능)
        Vector2 scroll = Mouse.current.scroll.ReadValue();
        Debug.Log($"Scroll Delta: {scroll}");
    }
}
```

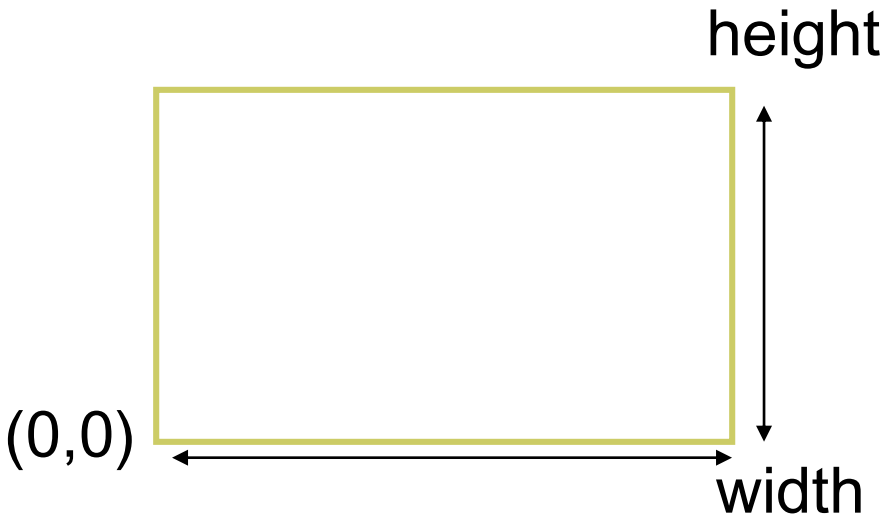
Mouse Event Callback

- Call this function from the **Update()** function, since the state gets reset each frame.

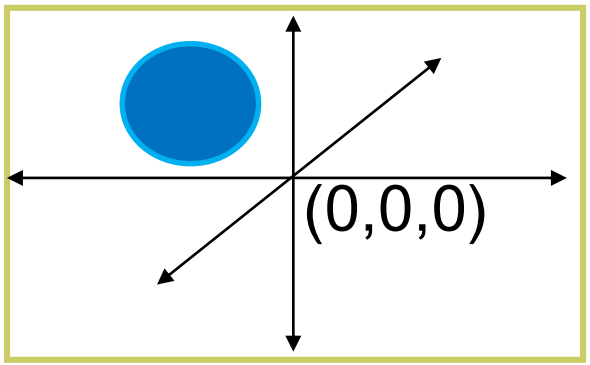
```
public class Example : MonoBehavior {
    void Update()  {
        if (Mouse.current == null) return;
        // The value is in the range -1 to 1
        float h = Mouse.current.delta.ReadValue().x * rotSpeed *
    Time.deltaTime;
        float v = Mouse.current.delta.ReadValue().y * speed *
    Time.deltaTime;
        // left-mouse holds to print the mouse position
        if (Input.GetMouseButton(0)) {
            Debug.Log(Input.mousePosition);
        }
    }
}
```

Mouse Positioning

- In Unity, the screen coordinate has the origin at the bottom-left corner, x+ is increasing to the right, y+ is increasing upwards.



2D screen coordinates



3D world space coordinates

Mouse Positioning

```
Vector3 worldPosition
// 2D mouse position -> 3D world position
void Update()
{
    // 2D Mouse Space – Unity6 New Input System (0,0) 좌하단 y+ 위 증가
    Vector2 mousePos = Mouse.current.position.ReadValue();
    // Screen Space
    Vector3 screenPos = new Vector3(mousePos.x, mousePos.y,
    Camera.main.nearClipPlane);
    // 스크린 좌표 -> 월드 좌표
    worldPosition = Camera.main.ScreenToWorldPoint(screenPos);
}
```

Mouse Positioning

```
private Vector3 worldPos;
private Vector3 screenPos;
void OnGUI() {
    // 2D Mouse Space – Unity6 New Input System (0,0) 좌하단 y+ 위 증가
    Vector2 mousePos = Mouse.current.position.ReadValue();
    screenPos = new Vector3(mousePos.x, mousePos.y, Camera.main.nearClipPlane);
    // 2D Screen Space -> 3D World Space
    worldPos = Camera.main.ScreenToWorldPoint(mousePos);

    GUILayout.BeginArea(new Rect(20, 20, 250, 120));
    GUILayout.Label("Screen pixels: " + Camera.main.pixelWidth + ":" +
    Camera.main.pixelHeight);
    GUILayout.Label("Mouse position (Screen): " + mousePos);
    GUILayout.Label("World position: " + point.ToString("F3"));
    GUILayout.EndArea();
}
```