Collaboration Strategies for Drag-and-Drop Interactions with Multiple Devices

Marc Davenport, Dalton Ott, Stephen Hughes
Coe College

Motivation

Multiple Input Devices

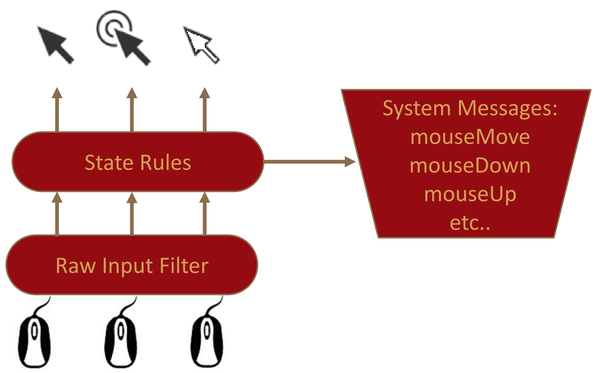
When multiple mice are attached to a single machine, the default response is to aggregate input to affect the position and behavior of a single mouse pointer. However, in collaborative environments, it may be desirable for each user to have their own pointer. Special software is needed to implement this.

The "Drag-and-Drop" problem

- "Dragging" is not a primitive/atomic system gesture; it is a temporal sequence of a mouseDown event followed by a mouseUp event.
- Issuing a mouseUp system message from any mouse causes *all* mice to drop; effectively only one mouse can drag at a time.
- Any system implementing multiple pointers will need to address this behavior.

ManyMouse Framework

- Custom software tool implemented at the system level
- Acts as a Proxy for mouse input:
  - Intercepts mouse input commands from all connected mice.
  - Allows for system input messages to be passed through to (or withheld from) any running application.
  - Changes display of mouse pointer to reflect current state.



Collaboration Strategies

Delegated

Delegated strategies attempt to select one of the connected mice to be in control of drag-and-drop activities on behalf of the group.

Inspired by traditional concurrency algorithms.

Trusted Authority:

- A "primary" device is selected and given to an authoritative operator. All drag-and-drop activity is controlled by this single user.
- Collaboration takes place verbally with requests for the operator to take action; the role of other mouse pointers is minimized.

Time out:

- Ability to drag-and-drop is rotated through users; attention to a particular input's mouse button actions expires after a set period of time.
- Enforces turn-taking, but may be difficult to tune the duration of an epoch.

First-Come, First Served

- Authority for mouse button listener is established by the first mouseDown event; mouse button events from other devices are ignored until the original device issues a mouseUp event.
- Serializes mouse button activity, and leaves input devices largely independent

Collective

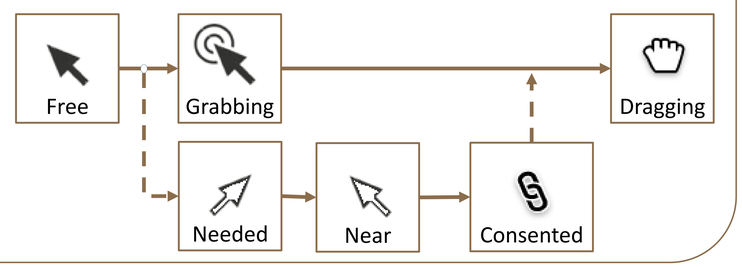
Collective strategies attempt to simultaneously include input from multiple devices to infer a coordinated effort from the group.

Inspired by the communication associated with coordination required to complete complex tasks.

Heavy Objects:

- Mandates collaboration by requiring drag-and-drop actions to have the support of a critical mass of participants; objects are "too heavy" to drag alone.
- All pointers are considered "Free" until the first mouse button is pressed. If the button is held down, it indicates that an object has been grabbed. The grabber needs to wait in that location until some collaborators come to help to drag the object elsewhere.
- Collaborators consent to a drag action by moving near the grabber and pressing the mouse down; they can withdraw their consent at any time by releasing the mouse button.
- Once a critical mass of participants consent, the initial grabber is allowed to move, dragging the selected object.
- The position of the consenters' mouse pointers move relative to the dragging pointer; this eliminates the need to synchronize movements.
- The implication is that the dragger ultimately decides where to drop, but they must have the consent of their collaborators to perform any repositioning occurs.

State	MouseDown	MouseUp	MouseMove
Free	Grabbing; Others to Needed	Ignored	Passed Through
Grabbing	N/A	Free (all)	Ignored
Needed	Ignored	Ignored	Passed Through
Near	Consented	Ignored	Passed Through
Consented	N/A	Near	Ignored
Dragging	N/A	Free (Drop)	Passed Through



COE COLLEGE

{shughes, mdavenport, ddott}@coe.edu