

The SPRING main loop

11-April-2007

SPRING is an interactive program, providing connections among:

- a. The *model* of SPRING's objects including tools, tissues, and geometric shapes
- b. Geometric input devices such as 3-D sensors, haptic devices, mice, and other interactive tools. These are collectively known as *sensors*.
- c. Output devices including graphics on displays, haptic force feedback, graphical tool values, and text.

In addition, SPRING uses settings from graphical user interface (GUI) controls such as menus, check boxes, type-in boxes, sliders, and other controls, collectively known as *widgets*. Some widgets also are used for display of output values in graphical or text form, including dials, graphs, sliders, progress bars, and others in the GUI.

SPRING's operation requires that it maintains its modeled world by continuous operation of the physics of the simulation, providing updated graphical displays and timely force feedback to haptics devices. The physical model requires most of the CPU time, and quick feedback of haptic forces is essential for users to maintain the "feel" of the model.

In addition, to maintain interactivity, SPRING must always react promptly to user-initiated changes in the GUI and to geometric input such as movements of 3-D sensors and pointing devices. Since many of the interactions with 3-D devices are through network connections, SPRING must also be responsive to network communications.

To achieve this, SPRING user two modes:

1. The main loop of the GUI toolkit, which handles user interactions with the graphical widgets such as menus, buttons, sliders, etc.
2. The Idle() event handler, which is called by the GUI's main loop whenever no outstanding events demand its attention.

The second is called very frequently, since most of the time the computer is idle. This frequent calling allows SPRING to give frequent updates to haptic devices to provide realistic forces.

Also, the main computation of the model's collisions between objects, forces between objects, deformations and changes of shape, and physical simulation are all performed in this loop.

The GUI's main loop is in control. The current version of SPRING is based on GLUT and GLUI, which in turn, are build on top of OpenGL. The main program first initializes interfaces, key objects, and sets up event handling, finally calling glutMainLoop(), which never ends.

Setting up event handlers

Last update 20-June-2006

The following calls set up event handlers for GLUT and GLUI. The first two callback functions manage changes to display windows. The Display function is triggered by an explicit call to `glutPostRedisplay()`. The Reshape function is called when the size of a window has been changed.

```
GLUI_Master.set_glutDisplayFunc(Display);  
GLUI_Master.set_glutReshapeFunc(Reshape);
```

These callbacks handle keyboard events, and are called when the key is released. The first handles standard keys, and the second deals with function keys, arrows, and *edit* keys such as Insert, Home, PageUp, etc.

```
GLUI_Master.set_glutKeyboardFunc(Keyboard);  
GLUI_Master.set_glutSpecialFunc(Special);
```

The three functions here handle mouse events. The first is called when a mouse button is pressed or released (GLUT_DOWN or GLUT_UP). `PassiveMotion` is called when no mouse button is selected. The third is called when the mouse moves with a pressed mouse button.

```
GLUI_Master.set_glutMouseFunc(Mouse);  
glutPassiveMotionFunc(PassiveMotion);  
glutMotionFunc(Motion);
```

This sets up the call when GLUT has no other events pending. Since human-triggered events are generally rare and quickly handled, most of the programs time is spent in the call to this `Idle()` function.

```
GLUI_Master.set_glutIdleFunc(Idle);
```

After these event handlers are initialized, the `glutMainLoop()` function is called, giving control to the event processing mechanism.

SPRING's Idle() function actions:

Except for the event handling describe above, all SPRING's operations are performed from calls in the `Idle()` function. Here is an outline of the SPRING actions:

- If voice processing is enable, check for and handle any voice commands.
- Next, handle the active sensors, which are graphical input and haptic devices. This involves:
 - Reading new 2-D or 3-D positions values – this is generally done via sensor servers, which communicate with SPRING via network sockets.
 - Sensing buttons and switches actuated on input devices
 - Coordinating and averaging the haptic input from cooperating haptic devices across the network.
 - Sending new force feedback to haptic outputs.
 - Updating the view of the simulated world if a sensor moves the system's *virtual camera*.
- Perform the behavior operations when tools such as forceps, cutters, and other tools interact with tissue objects.

- Update the *universe*, which applies the physics modeled to each object. Forces, velocities, and positions of each part of the physical world being simulated are recomputed using the spring interactions between nodes.
- Detect and resolve any collisions between objects. Resolution includes separating nodes when cutting, and calculating the haptic feedback to the user for the collisions.
- Update statistics on the simulation's operation.

This set of operations handles all the simulation computation and non-GUI input/output.

Handling the sensors (to be completed.)