New Features in OpenSim 3.1

In OpenSim 3.1, we've focused on improving the OpenSim scripting interface, accessible through the Graphical User Interface (GUI) and Matlab. We've also added to the OpenSim modeling and simulation libraries, with new components for modeling devices and estimating metabolic cost. OpenSim 3.1 also includes several usability improvements, including a toolbar for quickly running a forward simulation. As with every release, we've also implemented performance improvements and bug fixes. Improvements and additions include:

Enhanced scripting interface in Matlab and the GUI

Scripting through Matlab and the GUI allows users to call the full set of OpenSim modeling and simulation libraries (the API) without learning C++. Scripting also allows users to create batch scripts to streamline their workflows. We improved the OpenSim scripting interface for OpenSim 3.1. Enhancements include:

- Exposure of nearly all of the OpenSim API, including the most commonly used Simbody dynamics objects and functions, the Moment Arm Solver, and all the new modeling components described below
- Autocompletion in Matlab
- A streamlined Scripts menu in the GUI with shortcuts to find and run recent scripts.
- Enhanced error handling and reporting for script users
- Expanded plotting capabilities, including:
    - Plotting of threshold values
    - Plotting of fiber lengths, tendon lengths, etc. as a function of a motion
    - Ability to set labels on plots and get handle to plot windows after creation for additional editing

Learn more about Scripting in the User's Guide. Both new users of the Matlab scripting interface and 3.0 users need to follow the steps on Scripting with Matlab

Metabolic calculators

The metabolic calculators or probes, based on the work of Umberger and colleagues (2010) and Bhargava and colleagues (2004), estimate cost during motion for the whole body or individual muscles. Learn more about OpenSim Probes in the User's Guide and try them out in the example Simulation-Based Design to Reduce Metabolic Cost.

Expanded modeling toolbox

The toolbox is designed to improve OpenSim's ability to design assistive devices and other mechanisms. The new components can interface with the OpenSim workflow (e.g. computed muscle control) and are editable in the GUI. Where appropriate, they are visualized in the GUI and the API visualizer. The new components include:

- A spring that acts along a path (PathSpring)
- A clutched path spring (ClutchedPathSpring)
- An expression based bushing that applies torques as a symbolic function of deflections (ExpressionBasedBushing)
- An expression based point to point force that applies forces as a function of deflection and deflection speed (ExpressionBasedPointToPointForce)
- A planar joint (PlanarJoint)
- A gimbal joint (GimbalJoint)

You can find documentation for each of these components in the OpenSim Doxygen, look up how to define these components in the GUI's XML Browser, and see examples in the new tutorial Dynamic Walking Challenge: Go the Distance!.

Forward simulation toolbar in the GUI

The new toolbar for quickly running forward simulations in the GUI allows users to:

- Set coordinate speeds as initial conditions for a forward simulation
- Quickly run and stop a forward simulation using the simulation controls in the GUI Toolbar

Performance and accuracy improvements

- We added tighter assembly tolerance to enforce constraints.
- We've fixed several memory leaks throughout the OpenSim code base.
- We implemented several improvements to the new Millard muscles models to reduce computation time.

Additional Usability Improvements

- The XML browser in the GUI now maintains new lines when you copy and paste to an .osim model file in a text editor.
- Users can add and delete probes via the Navigator Window in the GUI.
- To simplify the setup process for reducing residuals, we've hidden the (almost always unnecessary/redundant) Control Constraints file from RRA settings. The examples have been updated to reflect this change and we issue a "deprecated" warning when a user runs RRA with a constraints file.
- Users can set max iterations and convergence tolerance of the optimizer for Static Optimization.

Additional API Improvements

- We created a new "advanced" interface for ModelComponents, consisting of overrideable realize() methods.
- Millard Muscle curves are OpenSim Functions to allow plotting via scripting.
- We added a method to set the name of the output file for the InverseDynamicsTool.
- Users can set muscle thickness and color in the API visualizer.

New examples

We've added to the library of OpenSim examples and tutorials. New materials include:

- A new simple gait model with 10 degrees of freedom and 18 muscles that uses the MillardEquilibrium muscle model
- Simulation-Based Design to Reduce Metabolic Cost
- Dynamic Walking Challenge: Go the Distance!
- Pulling Out the Stops: Designing a Muscle for a Tug-of-War Competition
- Sky High: Coordinating Muscles for Optimal Jump Performance
- Depreciated_CPP_From the Ground Up: Building a Passive Dynamic Walker Model

Bug fixes

- Fixed crash in Excitation Editor when creating new excitations for a new model
- Fix bug where excitation editor was losing changes upon load of a layout
- Fixed crash in Matlab/Scripting when creating Joints and adding them to model
- Fixed bug where ContactMesh loses associated geometry file on serialization
- Fixed bugs preventing loops from the GUI scripting shell
- Fixed GUI script handling of non-existing models
- Fixed bug where a Controller added to a model in the API was lost on serialization
- Enabled text and graphical editing of all actuators that use GeometryPath
- Fixed handling of coordinates out of range when computing geometry paths; prevents memory leaks
- Fixed bug in calculation of deactivation time constant for Thelen and Millard muscle models to reflect published paper
- Fix to Millard muscle models so steady state activation reaches the value of input steady state control
- Added option to exclude actuators from being controlled by CMC
- Removed obsolete Edit context menu from Actuator nodes in GUI
- Fixed slow plotting of Moments and MomentArms in the GUI plotter
- Removed duplicate defaults on round-trip read/save of objects by excitation editor.
- Fixed GUI handling of optional properties
- Fixed a bug in reporting of CMC and RRA results where analysis outputs were being linearly interpolated and then sampled
- Fixed a bug in StaticOptimization that was estimating target accelerations from double differentiated coordinates (q) after the model had been assembled
- Fix setting of initialStates in ForwardTool to be based on name rather than assumed (possibly incorrect) order.
- Fixed GUI tool issue with sizing of Time settings box
- Fixed bug in previewing experimental data