Engineers and researchers at the Robotics and Mechatronics Center (RMC) of Deutsches Zentrum für Luft- und Rundfahrt (DLR, the German Aerospace Center), are developing robots capable of learning about and interacting with their environment. Among them is Agile Justin, one of the most advanced two-armed mobile humanoid robots in the world. Agile Justin has 53 degrees of freedom, including 19 in its upper body, 26 in its hands, and 8 in its mobile platform. It senses the environment via stereo cameras and RGB-D sensors in its head, torque sensors in all joints, and tactile sensors on the skin of its fingers.

The DLR RMC team used Model-Based Design with MATLAB® and Simulink® to develop advanced control, calibration, and path-planning algorithms for Agile Justin. “Model-Based Design with MATLAB and Simulink covers a wide range of software domains needed for the design of advanced robotic systems,” says Berthold Bäuml, head of the autonomous learning robots lab at DLR. “It enables the simulation of complex mechatronic systems and controllers, code generation for real-time HIL testing, signal and image processing, and data analysis and visualization.”

The Challenge
Develop control systems for a two-armed mobile humanoid robot with 53 degrees of freedom

The Solution
Use Model-Based Design with MATLAB and Simulink to model the controllers and plant, generate code for HIL testing and real-time operation, optimize trajectories, and automate sensor calibration

The Results
- Programming defects eliminated
- Complex functionality implemented in hours
- Advanced control development by students enabled

The Challenge
Advanced humanoid robots need control systems that can quickly process input from a variety of sensors, plan continuous trajectories, and manage the movement of dozens of joints simultaneously.

Initially, DLR RMC engineers wrote separate control algorithms for the robot’s hands and arms manually in C/C++. When they started to create a single control loop for all degrees of freedom from the fingertip to the shoulder, they found that the system was too complex to code by hand.

DLR developed a new robotic software framework, aRDx, to help address this challenge, but they also needed tools that would enable them to automatically generate code from design models and perform hardware-in-the-loop (HIL) testing. Further, they wanted tools that postgraduate students could use to quickly develop their own robotic control systems.

The Solution
DLR RMC adopted Model-Based Design with MATLAB and Simulink for the development of advanced control systems for Agile Justin and other autonomous robots.

For individual joints, the team created a plant model using differential equations, which they incorporated into Simulink as an S-function.

They created a proportional-integral-derivative (PID) controller with Simulink and Control System Toolbox™, and tuned the control parameters via simulation.

Using Simulink Coder™ they generated C code from their controller model and deployed it to a PC running a QNX real-time operating system. HIL tests enabled them to compensate for sensor noise and precision as well as significant elasticities in the joint’s gears and nonlinearities in motor torque.
After verifying the controller design, the team generated C code for their production target, a Texas Instruments™ DSP. They then developed motion controllers for Agile Justin components with multiple joints, such as the torso and arms. To handle the more complex dynamics of these larger, heavier mechanisms, the team designed a Cartesian impedance control system in Simulink.

As with the single-joint controllers, the team tuned parameters in Simulink and generated code for HIL testing using Simulink Coder. During testing, the group used Stateflow® to sequence tasks such as grasping and lifting a basket.

They applied the sequential quadratic programming algorithms available in Optimization Toolbox™ to perform a constrained, nonlinear optimization for planning whole body motions—for example, to maximize the distance that Agile Justin can throw a ball.

The team developed automatic calibration algorithms in MATLAB, using Image Processing Toolbox™ to identify markers on the hands from images captured from the stereo and RGB-D cameras. The algorithms remove noise from the marker localization using a Butterworth filter from Signal Processing Toolbox™.

DLR is using Model-Based Design to develop new controllers that will increase Agile Justin’s ability to learn from its environment.

**The Results**

**Programming defects eliminated.** “Code generation with Simulink Coder has eliminated programming defects introduced when manually coding a control design in C/C+,” says Bäuml. “Functional defects have been reduced by about 80%, even though we are designing much more complex algorithms than we ever attempted with hand coding.”

**Complex functionality implemented in hours.** “For a throwing motion, Justin’s controllers coordinate 20 degrees of freedom to produce an optimal TCP trajectory,” says Bäuml. “We did this work in one afternoon using MATLAB, Simulink, and Optimization Toolbox.”

**Advanced control development by students enabled.** “With Model-Based Design, engineering students who work with us for only six months can develop sophisticated controls for a robot with 53 degrees of freedom because they don’t have to write code,” notes Bäuml. “Simulink enables them to cope with the interactions of multiple feedback loops and graphically construct and execute the complex algorithms.”

**Industry**
- Aerospace
- Industrial automation and machinery

**Application Areas**
- Control systems
- Mechatronics
- Robotics

**Capabilities**
- Data analysis
- Algorithm development
- System design and simulation
- Rapid prototyping

**Products Used**
- MATLAB
- Simulink
- Control System Toolbox
- Image Processing Toolbox
- Optimization Toolbox
- Signal Processing Toolbox
- Simulink Coder
- Stateflow

**Learn More About DLR Robotics and Mechatronics Center**
www.dlr.de/rmc/en