Physical Models in RoboChart

Pedro Ribeiro

robostar.cs.york.ac.uk

RoboChart: why a physical model?

For **simulation**.

- Using robotics simulators such as Gazebo.
- Explore different scenarios.

RoboChart: why a physical model?

For **simulation**.

- Using robotics simulators such as Gazebo.
- Explore different scenarios.

For **verification** of properties that involve physical aspects.

> *From takeoff, within 20s achieve and maintain an altitude with position accuracy of within 0.2m of the target altitude of 0.5m for a period of 3s.*

Such property relies on the combined behaviour of **software**, **sensors** and **actuators**.

RoboChart: physical modelling

Modular

- Blocks for **parts**, **links**, **joints**, **sensors** and **actuators**.
- Frame of reference determined by containment.

Record of assumptions

- Capture behaviour of **sensors** and **actuators**.
- Establish relationship with software model via **platform mapping**.

Tool support

- Eclipse-based tool that can automatically generate non-trivial SDF.

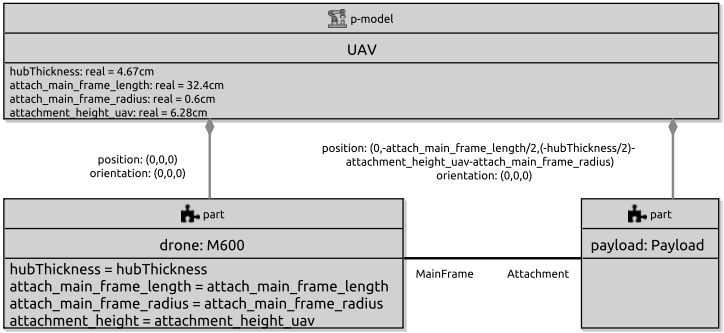
RoboChart: Firefighting UAV

Figure: DJI M600 with custom payload in flight.

1. Depth and thermal cameras.
2. Nozzle attached to two-axis gimbal.
3. Arduino for pump and gimbal control.
4. 1m-long carbon fibre arm.
5. LiPo battery.
6. 10bar water pump.
7. Onboard computer.
8. 4L water tank.
9. DJI M600 UAV.

Physical models: block diagrams

Block diagram contains a single **p-model** block that can be connected to other blocks.



Physical models: **links** and **bodies**

- A **link** is a representation of a rigid body and contains **joints**, **sensors**, **actuators** and **bodies**.
- A **body** captures the physical properties of a **link** but does not contain other elements.

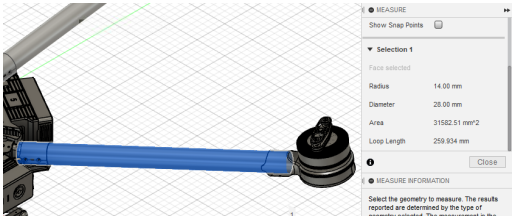


Figure: Rotor arm in M600 CAD model.

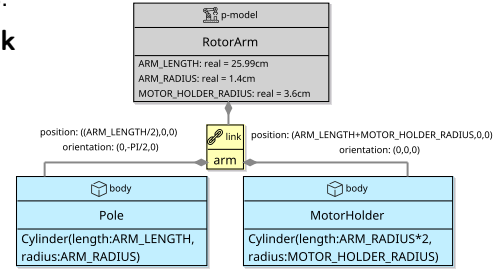


Figure: Physical model of rotor arm in RoboTool.

Physical models: **parts**

- **Parts** can be modularly composed in p-models.

  Their **links** can be connected to other components.

  Examples:
  - LeftLeg and RightLeg are LandingGear parts that reflect the left and right legs of the M600.
  - There are six arms in the M600, defined here using the indexed notation a : [1,arms] @.

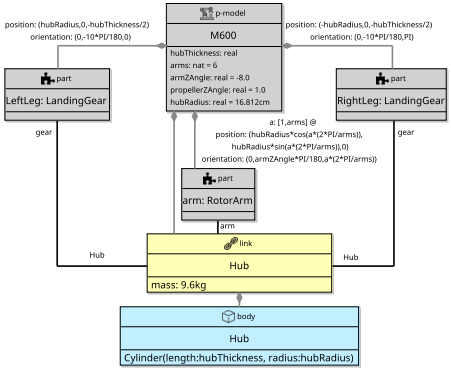


Figure: Legs and arms connected to the Hub.

Physical models: **joints**

- **Links** can be flexibly connected via **joints**.
  A joint is contained in a link and connected to another with a flexible connection.
- An actuated **joint** is defined by containing an instance of an **actuator**, such as a ServoMotor.

Figure: Pan and tilt servo actuators.

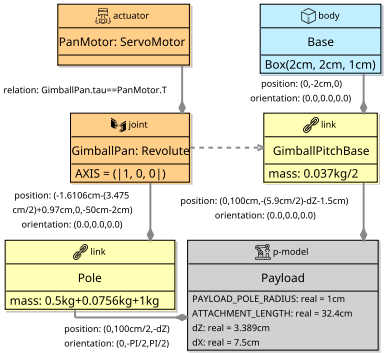


Figure: Partial model of spraying gimbal.

Physical models: **sensors** and **actuators**

- The behaviour of **sensors** and **actuators** is defined using a system of DAEs that define the relation between **inputs** and **outputs**.

  Example: in ServoMotor the input *dangle* is an angle and the output T is the torque produced.

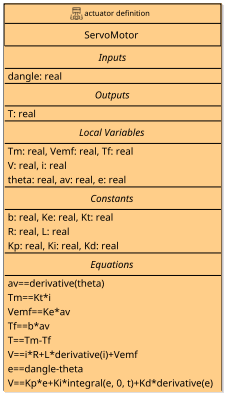


Figure: ServoMotor model.

Physical models: **sensors** and **actuators**

- Example: in ColourCamera the **input** is a 3D colour view of the world.

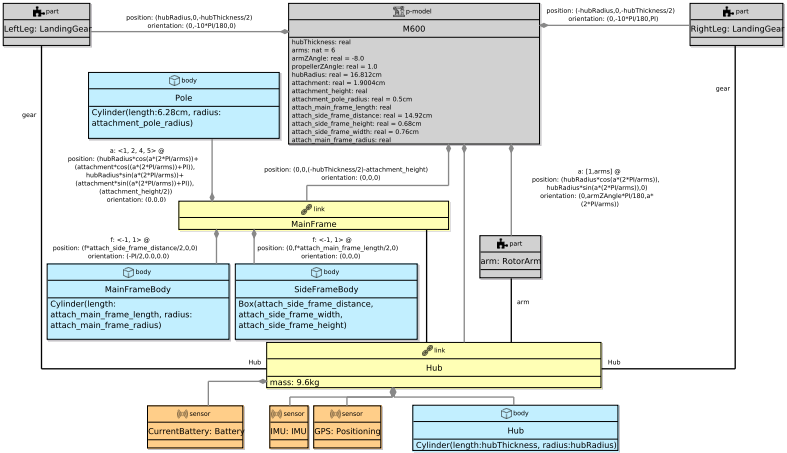
  The **output** is a 2D image produced following the camera's characteristics.

Figure: RealSense depth/RGB camera.

sensor definition
ColourCamera
Inputs
world: vector(real,3)->Colour
Outputs
image: matrix(Colour,960,720)
Local Variables
trans: vector(real,3)->Colour plane: vector(real,2)->Colour
Constants
WIDTH: nat = 960, HEIGHT: nat = 720 mx: nat, my: nat CM: matrix(real,3,4) = [|611.2014,0,321.56766,0; 0,611.63403,247.37758,0; 0,0,1,0|]
Equations
trans=={ op: vector(real,3) | op in dom(world) @ (|CM*hom3(op), world(op)|)} plane=={ p: vector(real,3) | p in dom(trans) @ (|dehom2(p), trans(p)|)} forall px: nat, py: nat | 1<=px^px<=WIDTH^1<=py^py<=HEIGHT @ image(px, py)==plane(mx*px, my*py)

Figure: ColourCamera model.

Physical models: complete M600 model



Physical models: **platform** mapping

- A **platform mapping** specifies how the software (**d-model**) and physical platform (**p-model**) are connected.

  Example: the platform operation gimbalWritePan is realised by assigning the value angle to the **input** angle of PanMotor as a radian.

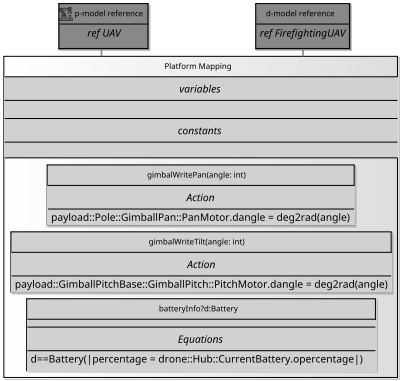


Figure: Subset of platform mapping for UAV.

Simulation: via SDF to Gazebo

RoboTool can generate a Scene Description Format (SDF) file suitable for simulation.

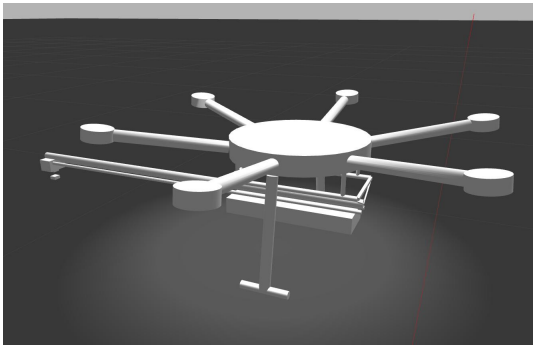


Figure: UAV loaded in Gazebo.

Concluding

- Brief overview of the physical modelling language.
- Example: Firefighting UAV.
- From modular block diagrams to simulation.
- Automatic generation of SDF suitable for simulation.
- Automatic generation of *CyPhyCircus* hybrid semantics for verification.
- Work is ongoing to describe scenarios suitable for model-based testing.