UAV Digital Twin via Probabilistic Graphical Models and ROS

Table of Contents

Description

This repository contains ROS 2 packages that implement dynamic structural health monitoring for a UAV via a digital twin imbued with a probabilistic graphical model.

This code is a companion to an academic research paper. If you use this work in an academic context, please cite the following publication(s):

Kapteyn, Michael G., Jacob V.R. Pretorius, and Karen E. Willcox. **A Probabilistic Graphical Model Foundation for Enabling Predictive Digital Twins at Scale.** arXiv preprint arXiv:2012.05841 (2020). https://arxiv.org/abs/2012.05841

```
    @article{kapteyn2020probabilistic,
      title={A Probabilistic Graphical Model Foundation for Enabling Predictive
Digital Twins at Scale},
      author={Kapteyn, Michael G and Pretorius, Jacob VR and Willcox, Karen E},
      journal={arXiv preprint arXiv:2012.05841},
      year={2020}
    }
```


Keywords: UAV, digital twin, graphical model, Dynamic Bayesian Network

Author & Maintainer: Michael Kapteyn, mkapteyn@mit.edu

Overview

digitaltwin package

This repository contains a ROS2 package called `digitaltwin`. This package implements all the core functionality of the dynamic in-flight health-monitoring demonstration.

This package is a demonstrative application that illustrates how a general purpose Bayesian Network software package (we here use Pomegranate) can be used to implement our proposed graphical model for digital twins. We use ROS in order to dynamically build and leverage the graphical model, and illustrate how information is transferred between an asset and its digital twin.

At its core, the code is a simple two node ROS architecture:
 ROS architecture for the asset-twin system

The `uav_asset` node is a simulated UAV asset. Its role is to acquire control inputs from the digital twin, progress the simulation one timestep, simulate sensor data accordingly, then pass this sensor data to the digital twin node.

- Functionality for the UAV is implemented in `src/digitaltwin/digitaltwin/UAV.py`

The `uav_twin` node is the UAV digital twin. Its role is to acquire sensor data from the UAV asset, decide which control input to issue, then pass this control input to the UAV asset. The digital twin does this by maintaining a probabilistic graphical model of the asset-twin system, as described in the accompanying paper.

- Functionality related to planning is in `src/digitaltwin/digitaltwin/planner.py`
- Functionality related to the probabilistic graphical model is in `src/digitaltwin/digitaltwin/graphicalmodel.py`. In order to better understand how the graphical model is constructed and utilized, it is greatly encouraged that you first read and understand the Pomegranate Bayesian Network documentation.

Visualization packages

In addition to the core functionality, this repository also contains six additional rqt_gui plugins. These plugins allow you to visualize various quantities as the simulation progresses:

- **rqt_state**: Plots the digital states, z_1 and z_2, versus time
- **rqt_trajectory**: Plots the digital states in state-space (i.e. z_1 vs. z_2)
- **rqt_sensor**: Plots sensor data and quantity of interest (i.e. reference sensor data)
- **rqt_control**: Plots control inputs vs. time
- **rqt_reward**: Plots reward functions vs. time
- **rqt_network**: Plots the probabilistic graphical model nodes and edges

Installation

Building from Source

Dependencies

These packages have been tested under ROS2 Eloquent on Ubuntu 18.04.

Dependencies include:

- Robot Operating System 2 (ROS2) (middleware for robotics),
- Python 3 (and standard packages such as `numpy`, `json`, `re`, etc.)
- Pomegranate 0.13.0 (Python library used for Bayesian Network back-end)

Building

To build from source, clone the latest version from this repository (or unzip the file contents) into a ROS2 workspace (here called `digitaltwin_ws`) and compile the package using

```
cd digitaltwin_ws/src
# [git clone/unzip file contents here]
cd ../
colcon build
```

You should now have the directories `digitaltwin_ws/install` and `digitaltwin_ws/build`

Usage

Run the demonstration with

```
cd digitaltwin_ws
. install/setup.bash
ros2 launch digitaltwin digitaltwin_launch_gui.py
```

To run the visualizations open `rqt_gui` while the `uav_asset` and `uav_twin` (should open automatically using the provided `*_gui*` launch file)

```
ros2 run rqt_gui rqt_gui
```

You can load a provided GUI layout by going to `Perspectives->Import` and importing the file `src/digitaltwin/digitaltwin.perspective`. This perspective should show the probabilistic graphical model, sensor data, states, and reward functions.

Alternatively, you can build a custom GUI by going to the `Plugins->Visualization` menu and selecting the plugins you wish to add to the GUI.

Config files

- **inputfiles/UAVconfig.json** Contains information related to the UAV use-case.

Note that this input file contains a series of pre-computed strain values computed using our structural model of the UAV, which is used under license and is unable to be shared publicly. Thus, reading strain values from this files should be viewed as a placeholder for a structural model evaluation. The strain values are stored in dictionary format as follows:

```
uav_config["observations"][z_1][z_2][control][e_value]
```

where

- $z_1 \in \{0, 20, 40, 60, 80\}$
- $z_2 \in \{0, 20, 40, 60, 80\}$
- control $\in \{2g, 3g\}$
- e_value $\in \mathbb{R}^+$ (30 samples are drawn from the posterior distribution of e)

Launch files

- **digitaltwin_launch.py:** Standard simulation. Spis up asset and digital twin nodes.
- **digitaltwin_launch_gui.py:** Simulation + Visualization. Spins up asset, digital twin, and rqt_gui nodes.
- **digitaltwin_launch_gui_logger.py:** Simulation + Visualization + Logging. Spins up asset, digital twin, rqt_gui, and logger nodes.

Nodes

uav_asset

Acquires control inputs from the digital twin, progresses the simulation one timestep, simulates sensor data accordingly, then passes this sensor data to the digital twin node.

Subscribed Topics

- **`/control_data`** ([digitaltwin/ControlA])

  The control input issued by the digital twin.

Published Topics

- **`/sensor_data`** ([digitaltwin/Sensor])

  Simulated sensor data generated by the UAV asset simulator.

uav_twin

Acquires sensor data from the UAV asset, decides which control input to issue, then passes this control input to the UAV asset.

Subscribed Topics

- **`/sensor_data`** ([digitaltwin/Sensor])

  Sensor data received from the UAV asset.

Published Topics

- **`/control_data`** ([digital_twin/ControlA])

  The control input issued by the digital twin.

Further Reading

Kapteyn, Michael G., Jacob VR Pretorius, and Karen E. Willcox. “A Probabilistic Graphical Model Foundation for Enabling Predictive Digital Twins at Scale.” arXiv:2012.05841 (2021). https://arxiv.org/abs/2012.05841

Contact

Please report bugs to **Michael Kapteyn, mkapteyn@mit.edu**.