Roscon + iROS 2019 Macau

About Mov.ai

- Helping robot developers
  - Quick goto market + long-term support
  - Visual launch of ROS Node network, in-browser IDE, SW Distribution, reversible upgrades, multi-protocol event processor
- Helping industrial operators and automation integrators
  - Tools to set up fleets of robots of different types

Commercial Grade ROS | ROS for Business

- VC funded startup, since 2016
- Team developed custom autonomous mobile robots for 7 years
- 20 engineers, based in Lisbon
- 2 AGV partners - cart moving (TUGBOT) and pallet moving (RPM)
- Launching in Macau ROSCON 2019

Inviting ROS community to join MOV.AI the Beta program

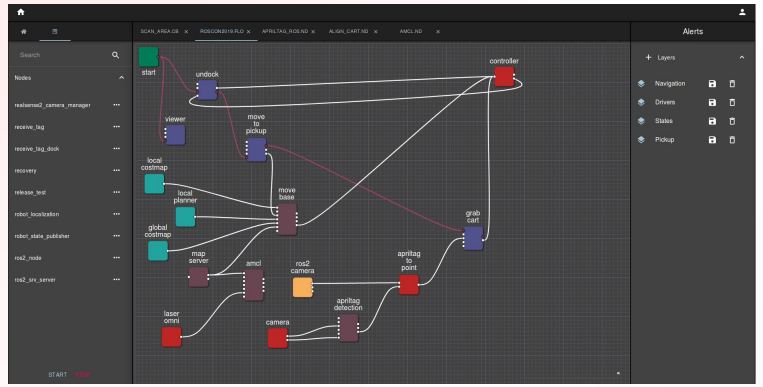
Visual Launch System for ROS

- Replaces roslaunch / rosrun framework with **V**isual **L**aunch **D**iagrams
- Drag & Drop Nodes, Connect Node inputs/outputs
- **VLD** lines represent communication protocols between nodes
  - Modify Nodes/connections in split seconds
  - Organize multiple Node networks, Node versions & Parameters
  - Supports all ROS protocols, TF, Nodelets, pluginlib

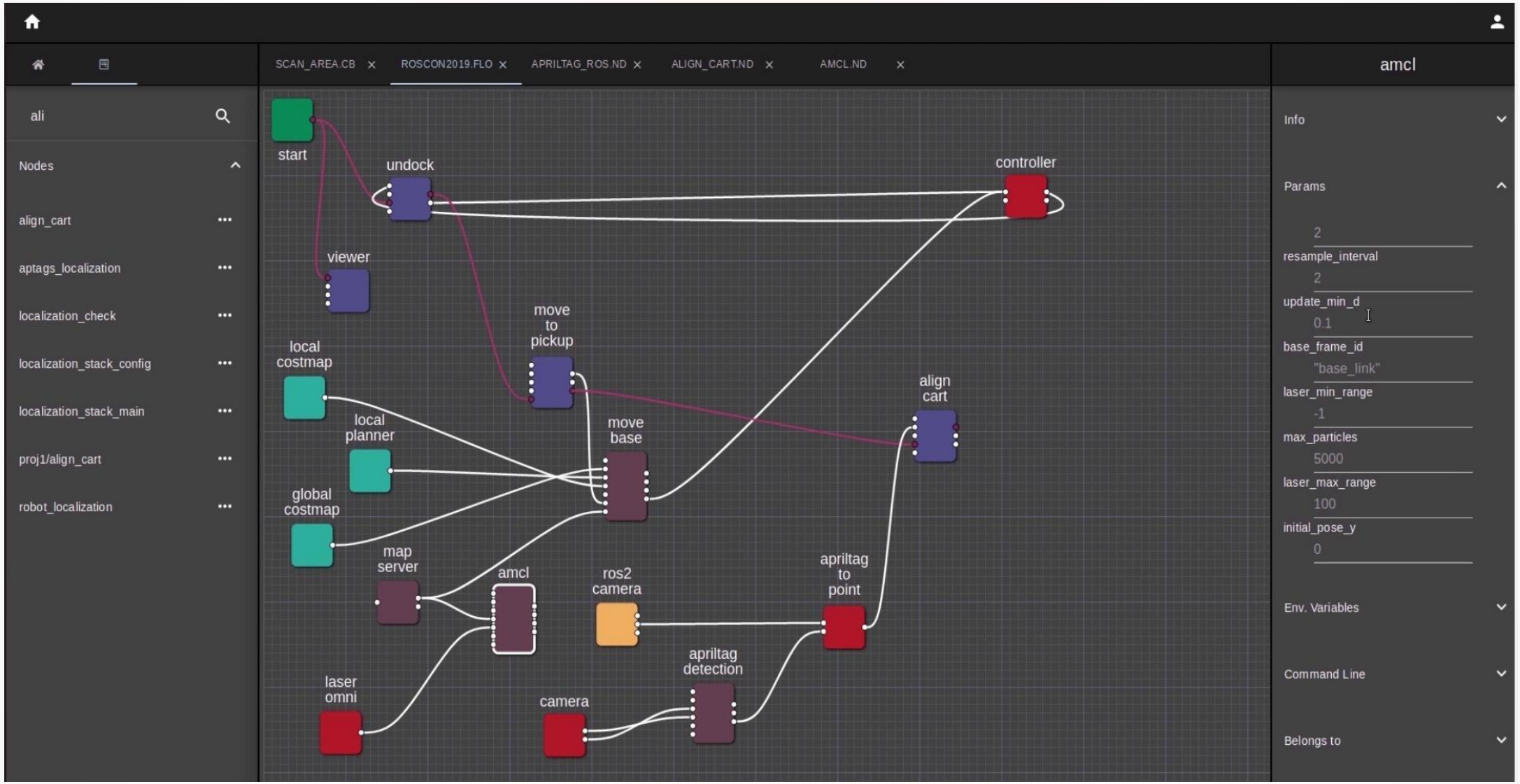
```
<launch>
  <!-- Turtlesim Node-->
  <node pkg="turtlesim" type="turtlesim_node" name="sim"/>
  <node pkg="turtlesim" type="turtle_teleop_key" name="teleop"
output="screen"/>
  <node pkg="learning_tf" type="turtle_tf_listener.py"
        name="listener" />

  <node name="turtle1_tf_broadcaster" pkg="learning_tf"
type="turtle_tf_broadcaster.py" respawn="false" output="screen" >
    <param name="turtle" type="string" value="turtle1" />
  </node>
  <node name="turtle2_tf_broadcaster" pkg="learning_tf"
type="turtle_tf_broadcaster.py" respawn="false" output="screen" >
    <param name="turtle" type="string" value="turtle2" />
  </node>

</launch>
```



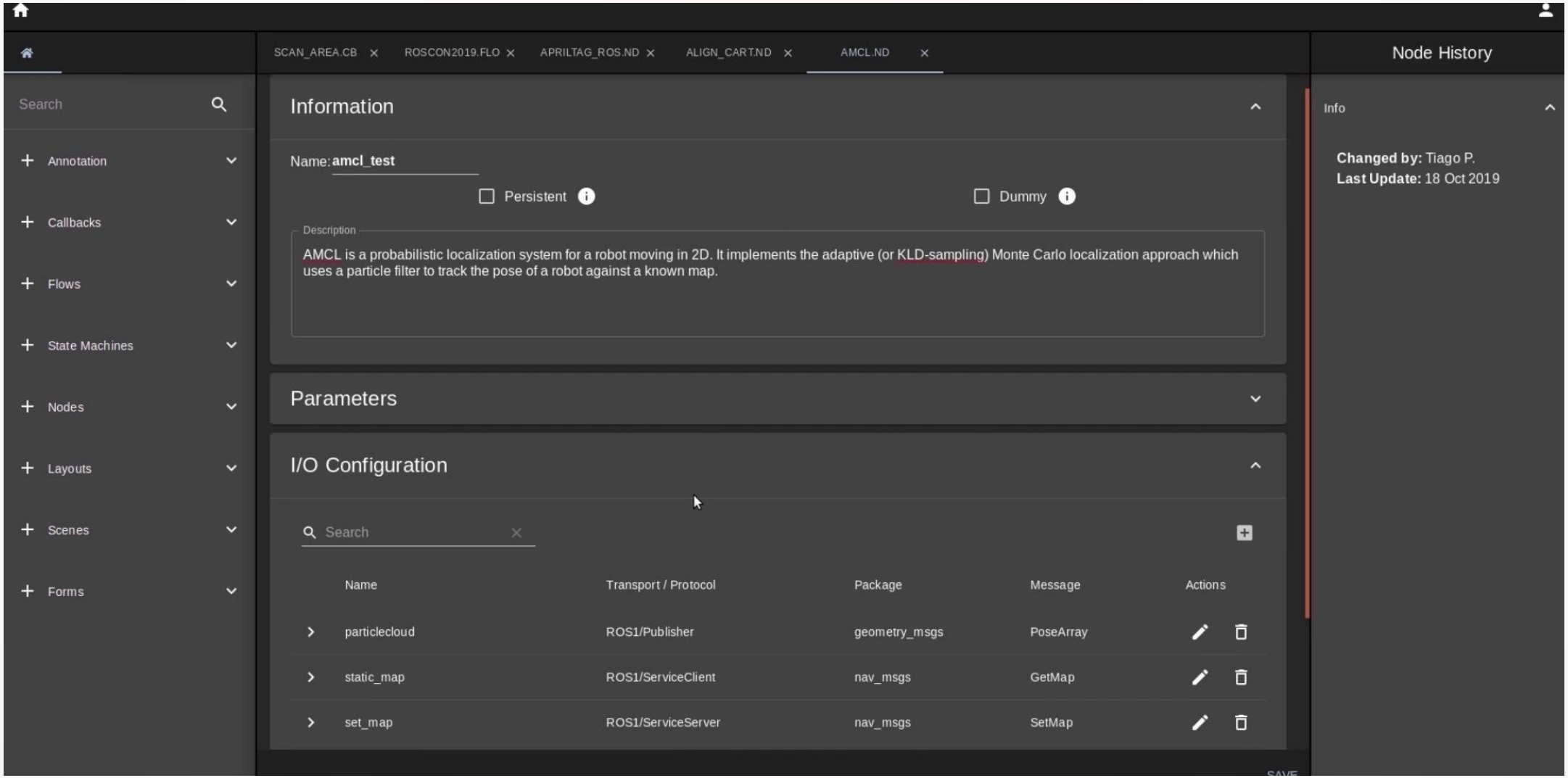
Visual Launch System for ROS



Node Setup for Visual Launch

- Data required to deploy existing ROS node in Visual Launcher
- Input / Output ports (pub, sub, action, service)
- Parameters - command line, parameter server, environment vars
- Default parameters (template), drag into a Flow (instance)
- Support for multiple versions of same Node
- Migration tools - to help in input all the Node related data

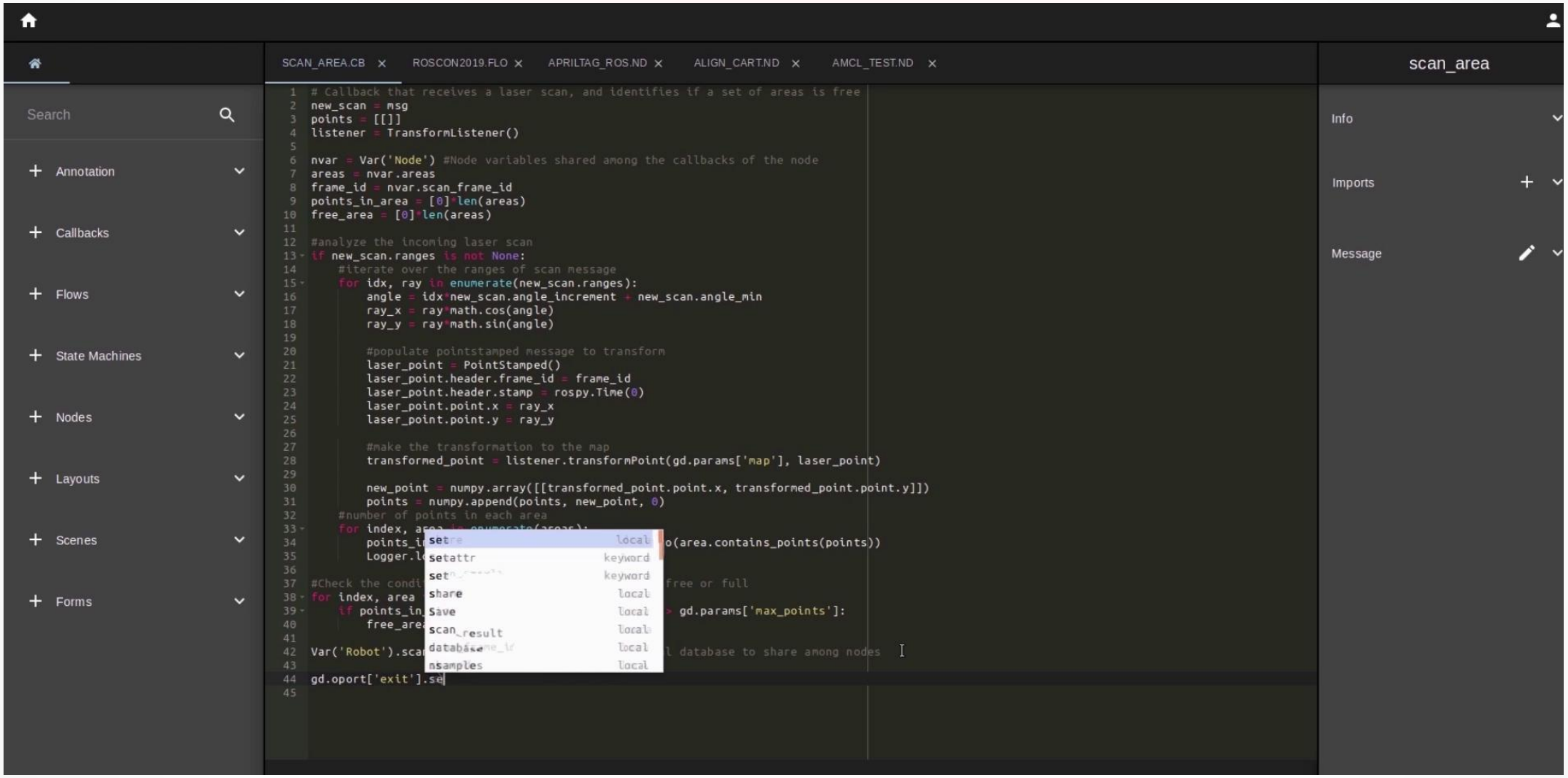
Node Setup for Visual Launch



MOV.AI Nodes - IDE in Browser

- Multi-Protocol Event processor
  - Message/Event triggers Callback in Python
  - Support for ROS1, ROS2, HTTP, WebSocket, Serial Driver, Redis DB..
  - Callback code cannot access communication layer
- Native parallel processing
  - Callbacks are Re-entrant - Persistent data only via Redis DB API
  - AsyncIO backend + Cython (C level performance)
  - Resource Usage Profiling tools
- Upgrade / Downgrade - mandatory for industrial clients
  - Imported libraries - outside the callback code
  - GIT based versioning of Callbacks

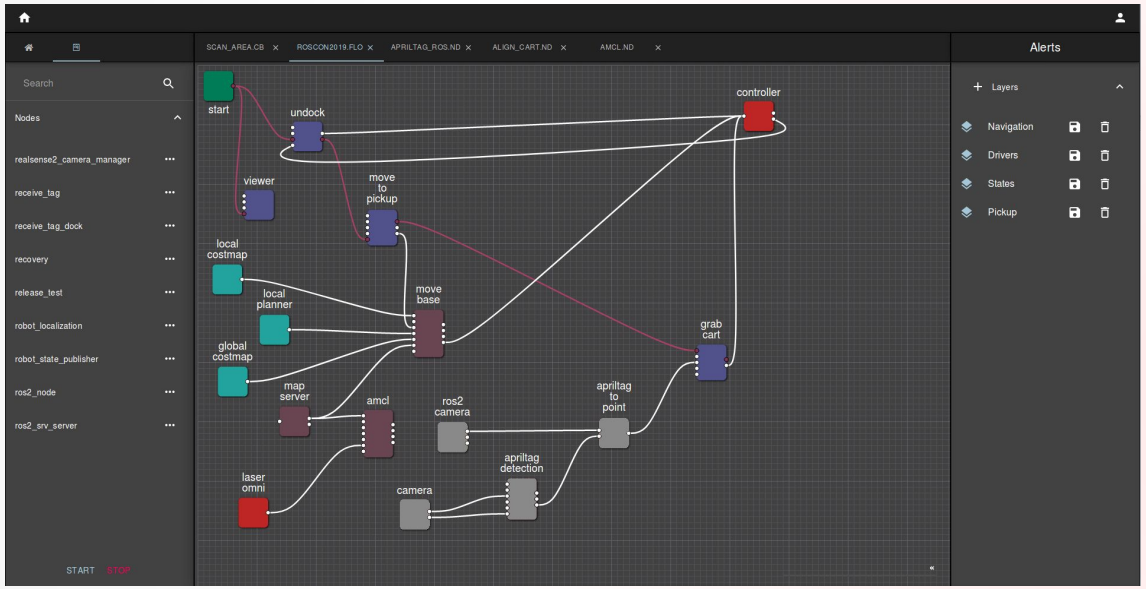
MOV.AI Nodes - IDE in Browser



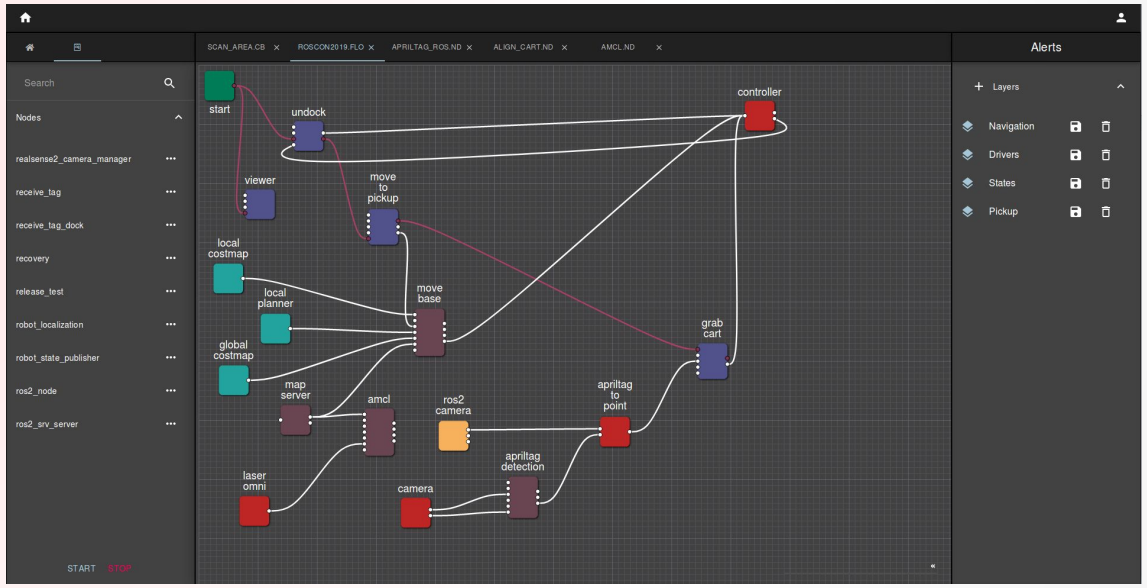
State Transition

- Visual Launch Diagrams can act as State Machine Diagrams
- MOV.AI Nodes can act as “State Nodes”
  - “Transition” is one of the supported VLD protocols
- When a Node is transitioned-to
  - All non-connected nodes are recursively killed/disabled
- Visualize dependency between Robot’s State & required ROS nodes
- Visual ROS2 lifecycle manager

State Transition



Corridor navigation state

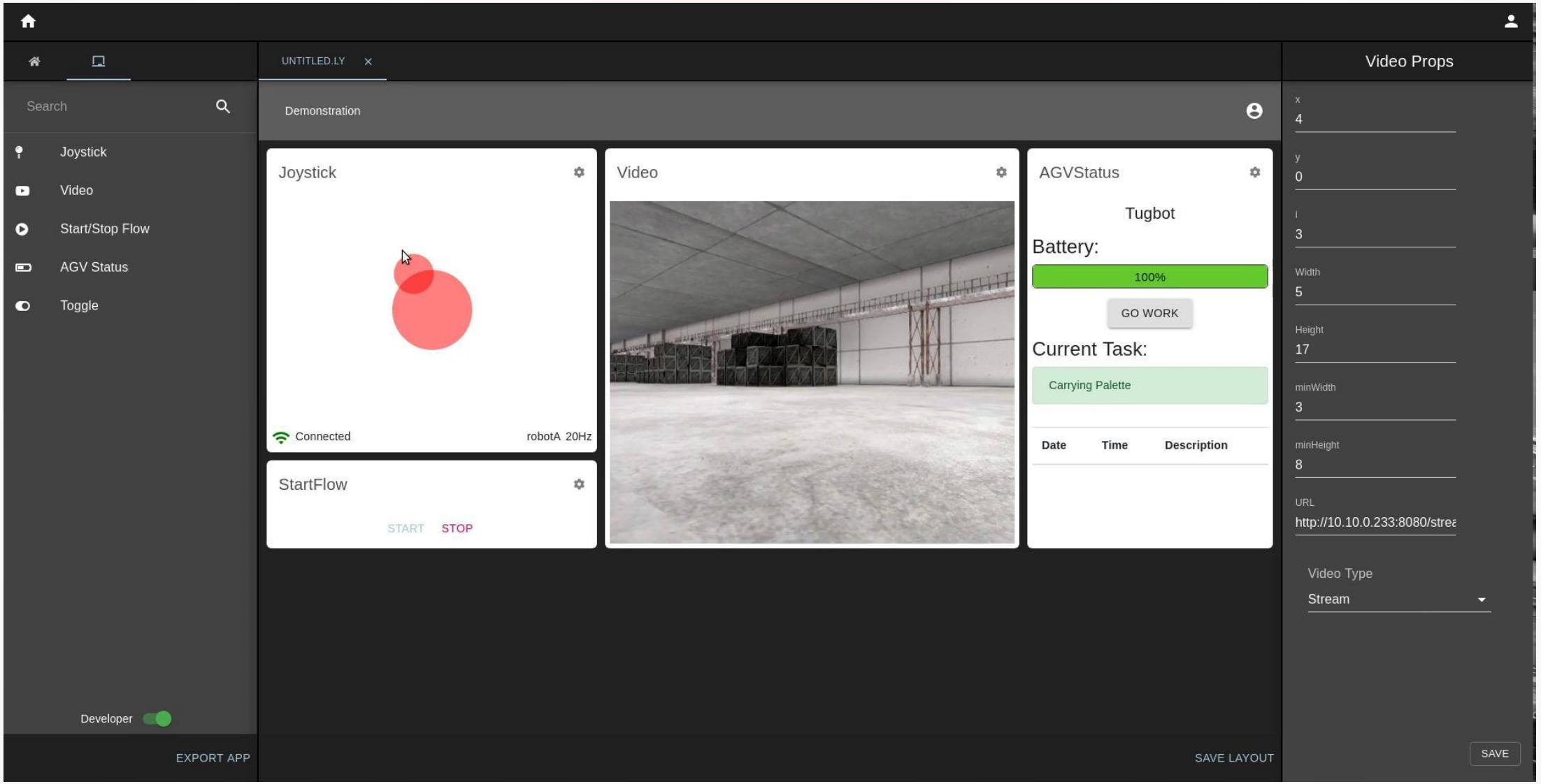


Cart Grabbing state

Customizable UI

- MVC framework
  - Collaborative UI (2-direct. link w db replicated on every robot)
  - Any Robot can act as Web server
- Extensible web API
  - HTTP/WebSockets protocols supported in MOV.AI Nodes
  - Custom server-side functions
  - REST API available for CRUDE actions (vs replicated DB)
- REST-full application
  - Develop apps with preferred js framework (React, Vue, etc)
  - Create and upload your own js application into Mov.AI system
- Dashboard Creator
  - Create operator views with stats and queues
  - Create custom dashboards to monitor and control fleets
  - Extensible set of customizable widgets

Dashboard Creator



Thank you!

www.mov.ai
info@mov.ai