Traffic network of Dubins cars as a benchmark

Scott C. Livingston Vasumathi Raman

***Abstract*—The proposed benchmark involves vehicles with differential drive dynamics operating in a network of multi-lane roads. As is well known, these dynamics can be further constrained so as to follow Dubins curves, hence the name of this benchmark. The proposal is for a domain of problems that begins with simply requiring a controller to steer a single robot to repeatedly visit a collection of waypoints while avoiding collisions with other vehicles and obstacles. This basic setting is made to be more sophisticated by adding rules of the road, such as preferred sides on which to drive and requirements for negotiating multi-stop intersections. Other vehicles are controlled by adversarial agents and may be made to follow certain templated behaviors, such as repeatedly driving around a loop or following behind the ego robot. Assumptions about the adversaries are expressed through temporal logic specifications, thus leading to reactive synthesis problems. The authors organized a competition at the Int'l Conference on Robotics and Automation (ICRA) during May 2016 that included trials in simulation for this benchmark. The messaging framework and package management are provided through ROS, and multi-robot physics simulation is achieved through Gazebo.**

I. Introduction

We present a benchmark that concerns formal synthesis for multi-robot motion planning tasks. It uses ROS (http://www.ros.org/) as the main framework for message-passing, trial management, and infrastructure for building and running packages. The benchmark is a problem domain. Particular problem instances are obtained by selecting values of various parameters and deciding whether a simulation engine or physical testbed should be used. Simulation is realized using Gazebo [1] (http://gazebosim.org/).

This benchmark is developed as part of a suite of benchmarks and competitions on formal methods for robotics (https://fmrchallenge.org). The first such competition was held in May 2016 and co-located with the Int'l Conference on Robotics and Automation (ICRA), where participants tried earlier versions of this benchmark.

II. Problem domain: Traffic network of Dubins cars

This domain involves navigation in a small network of two-lane roads with vehicles that follow unicycle-like dynamics. Now there is an adversary that selects the motion of other cars. The adversary is subject to assumptions such as obeying traffic rules and not parking unfairly at locations that must be eventually reached by the controller.

Note that some text here is copied or derived from text in the competition specification, which is available at https://fmrchallenge.org/norm/

A. Dynamics and constraints

All cars including the controlled robot are assumed to have the same rigid body shape and to have the same dynamics. Let $\mathcal{I}$ be the index set for the cars in the network, with r denoting the index of the controlled robot; all cars indexed with $i \in \mathcal{I}\backslash\{r\}$ are regarded as a part of the (adversarial) environment.

The pose of each car $i \in \mathcal{I}$ is specified by (x_i, y_i, θ_i), where $(x_i, y_i) \in \mathbb{R}^2$ is referred to as *position*, and $\theta_i \in S^1$ is referred to as *orientation*. The car's body is a rectangle or a circle, and the position is defined to be at the mean point (center of mass) of the body. Let w be the width of the car – this is identical for each $i \in \mathcal{I}$. The cars have unicycle dynamics.

The workspace is a randomly generated road network constructed as follows. Create a planar graph $G = (V, E)$ in which each vertex v has degree $d_v \leq 4$ (i.e., at most 4 neighbors). Embed this graph in the plane, and expand the edges to have width $4w$ (recall w is the common vehicle width).

For this aspect of the "traffic network of Dubins cars" problem domain, we are able to vary the bounds on the permissible control inputs, the size ($|V|$) and topology ($|E|$) of the road network, and the number of other cars, i.e. $|\mathcal{I}|$.

B. Specifications

Because there is now an environment that may behave adversarially, task specifications will be of the form $\varphi_{\text{env}} \Rightarrow \varphi_{\text{sys}}$, where φ_{env} is known as the "assumption" and φ_{sys} as the "guarantee." The desired behaviors to be realized by the robot are provided through φ_{sys} and include

1) obstacle avoidance while repeatedly visiting regions of interest, as in
$$q(0) = \text{init} \wedge \square\,(q(t) \notin \text{Obs}) \wedge \bigwedge_i \square \diamondsuit \text{goal}_i \qquad (1)$$
2) remaining in the right-lane except when it is blocked;
3) stopping at intersections, which are treated as all-ways stops, and then proceeding based on order of arrival.

The other cars will be assumed to follow the same road rules as listed above. However, exceptions (violations) may occur, and thus additional fairness assumptions will be provided. In particular,
$$\bigwedge_i \square \diamondsuit \texttt{free}(\text{goal}_\text{i}) \qquad (2)$$
which provides that other cars will always eventually vacate the i-th goal region. The position of the other cars will be provided when in a predetermined radius, to simulate solving the associated vision problem.

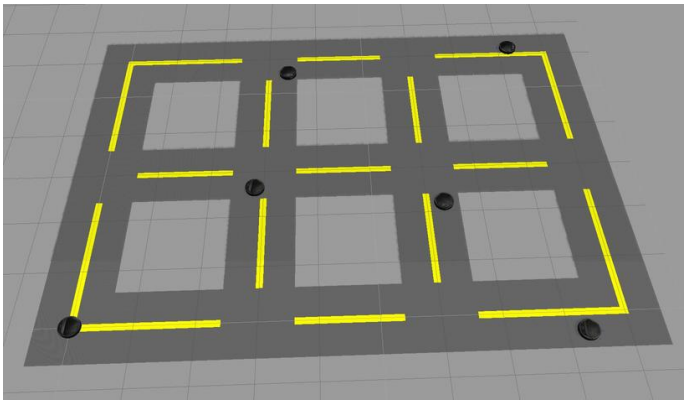

Fig. 1. Example of a road network

C. Controller Execution

Controller execution has the following key components:

- A *problem generation engine* produces random problem instances from a configuration file that includes information about the e-agents, in particular their number and namespace. Problem instances are in a format that uses JSON (http://json.org) as container and as demonstrated by the following code sample:

```
{
"version": 0,
"goals": [
[2, 0]
],
"rnd": { "version": 0, "length": 3,
"transform": [0, 0, 0],  "shape": [3, 4],
"segments": [[0, 0, 1, 0], ...] },
"intersection_radius": 1
}
```

  where rnd is a road network description consisting of segments, and $intersection\_radius$ is the distance from the end of a segment at which the vehicle is considered to be in the intersection. An example of a road network is shown in Figure 1, which is a screenshot capured from the Gazebo client.
- A *trial runner* runs the problem generation instance multiple times to generate several trials. The problem instances are published on the topic `probleminstance_JSON`. The trial runner subcribes to the model states in topic `gazebo/model_states` and publishes the labeled output to `loutput`.
- The form of the controller is left flexible, but it must do the following:
  - listen for problem instances by subscribing to the `probleminstance_JSON` topic
  - send control inputs and receive sensor measurements on the relevant topics for the vehicle, e.g., `/ego/mobile_base/commands/velocity` or `/agent0/odom/`.

D. Evaluation

E. Competition at ICRA 2016

In the competition at ICRA 2016, the score for each trial was decided as follows.

1) -1 if a collision occurs between the ego vehicle and any other vehicle;
2) otherwise,

$$\frac{(\text{\#e-agents}) * (\text{\#road grid rows}) * (\text{\#cols}) * X}{\max\{1, (\text{\#seconds ego vehicle not on some road})\}},$$

where $X = \#$unique visits to all goal intersections.

The number of trials, and parameters like the size of the road network will be the same for each team/controller.

A team's final score is the sum across all trials.

F. Extensions

Ongoing work (not used at ICRA 2016) is exploring more detailed evaluations, including the following criteria.

1) time to begin, i.e., duration between being first given the problem instance description and requesting start of the play.
2) number of collisions.
3) fraction of goals visited
4) all of the above with varying:
   a) numbers of e-agents;
   b) size of the road net;
   c) lane width.

III. Conclusion and Future Work

In this abstract we briefly present a benchmark for multi-robot control synthesis. Access to the source code, further details, and examples can be obtained via the following URLs:

1) https://fmrchallenge.org/#dubins_traffic
2) http://docs.fmrchallenge.org/en/v0.0.4/dubins_traffic.html
3) https://github.com/fmrchallenge/fmrbenchmark/tree/v0.0.4/domains/dubins_traffic

A basic realization of the benchmark in simulation was used in the first competition on formal methods for robotics at ICRA in May 2016. Development of the benchmark is ongoing, including prototypes for physical implementations based on the Kobuki mobile robot and Hokuyo laser range finder. Future work will include CloudSim (https://cloudsim.io/) or some other Web-accessible process for controller evaluation. Support will be provided for other communications libraries that are used in robotics research and industrial practice, in particular LCM (https://lcm-proj.github.io/).

References

[1] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Proceedings of the 2004 IEEE/RSJ Int'l Conference on Intelligent Robots and Systems (IROS)*, 2004.