Gmail **Luciano Barreira <luc.s.barreira@gmail.com>**

Re: Greetings from Brazil

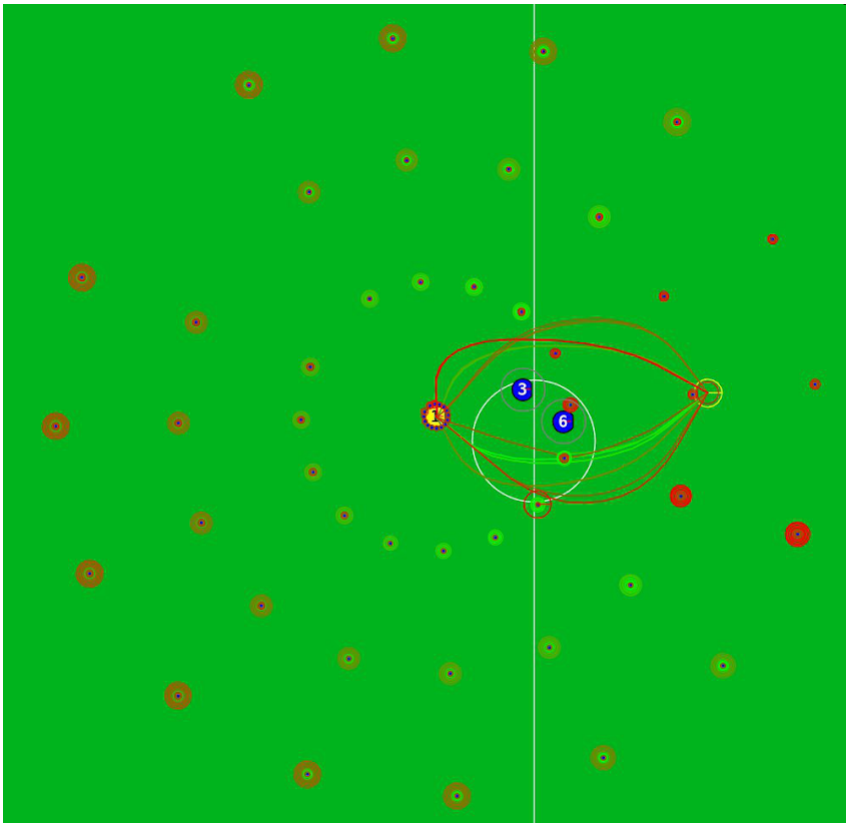
1 mensagem

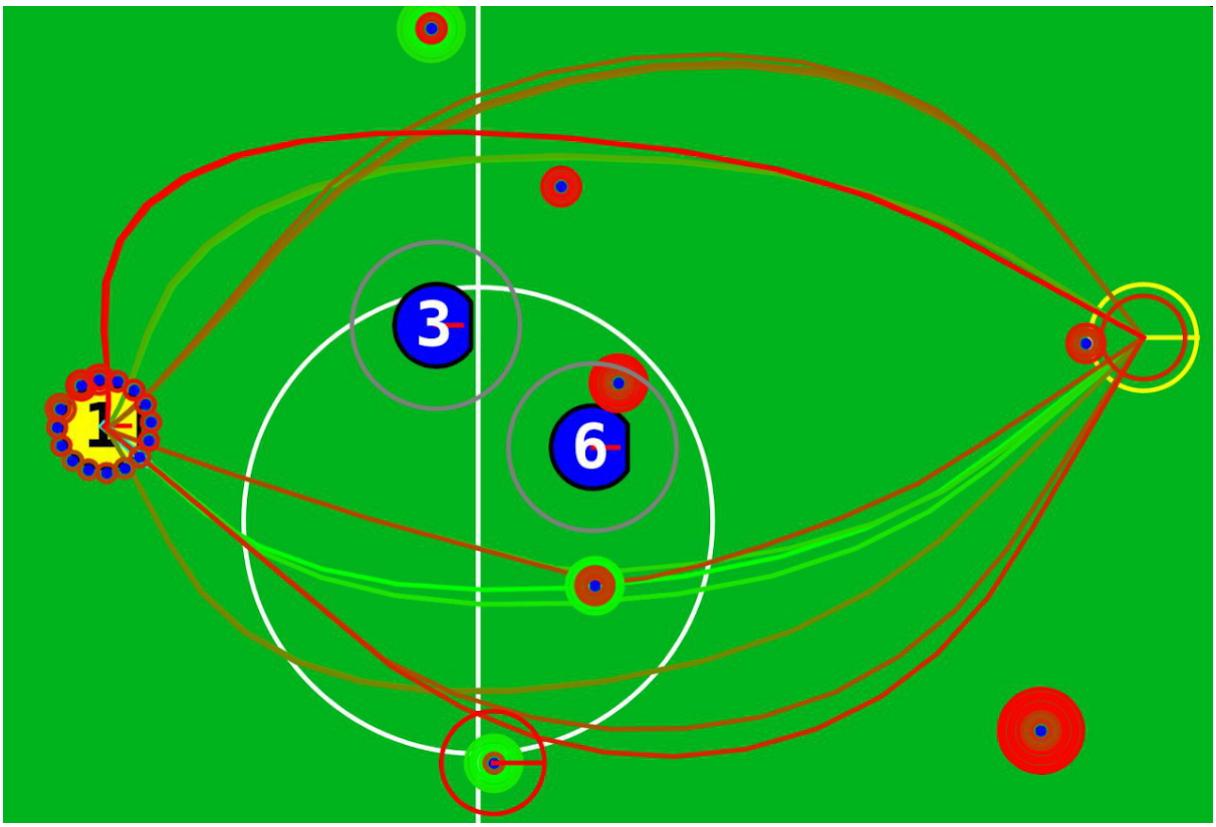
Andre Ryll <andre@ryll.cc> 27 de fevereiro de 2018 15:40
Para: luc.s.barreira@gmail.com
Cc: management@tigers-mannheim.de

Hi Luciano,

congratulations for winning the LARC and thanks for the warm words. It always feels great to receive an acknowledgement for our hard work and publications.

Concerning your questions:
Actually, we don't use any RRT or geometrical path planner. Our path planning works directly with bang-bang trajectories. The rough flow of the algorithm is:
1. Generate sub destinations in a star shape (see image)
2. Generate a trajectory to each sub destination
3. Step along this trajectory with a fixed step size (time on the trajectory)
4. For each of those steps, generate another trajectory starting on the location of the first trajectory and ending on the final destination
5. Combine both trajectory to a potential trajectory path
6. Determine collisions on each of those potential trajectory paths
7. Sort them by total time (and some score for collisions, in case there is no path without a collision)
8. Take the best





- Each blue dot is a sub destination
- The green/red circles around it represent the score for the path with this sub destination
- The radius of this circle represent the step on the first trajectory (smallest circle -> first step)
- Green is good, red is bad
- Grey circles are the considered obstacles (the actual size depends on the velocity on the trajectory)
- yellow circle is the final destination
- red circle is the chosen sub destination (which is currently sent to the robot)
- the best paths are painted in green to red

A path is recalculated on each AI frame (we currently try to achieve 100fps, independent of camera rates). We don't prioritize any specific robots, the path planning is fast enough to be done for all robots. As the way to the target is fully specified by a trajectory we can simple take the total time of the trajectory to estimate our arrival time. If there is a situation where we cannot reach the target, we will usually drive around a little crazy. This comes from the fact that we are only checking the first few seconds of the trajectory for collisions. If the intermediate position is far away and not in the direction of the target it will certainly have no collision for some seconds. This doesn't get you to your target, but sooner or later the situation on the field will change.
The target position or intermediate position is send to our robots. Internally, they also generate a trajectory to the target based on the internal state and follow that trajectory. This also detaches the robots a little from our AI and helps to mitigate delays.

The basic layer of our camera filter is quite simple. We also run a Kalman Filter for each robot on each camera individually. In fact its two Kalman Filters for a single robot, one for orientation (1D) and one for position/velocity (2D). The 2D filter is a simple Tracking Filter with constant velocity model. Merging a robot that is visible on multiple cameras is indeed a little more tricky. Fortunately, due to the Kalman filters we can actually merge them in a statistical ideal way. We simply weight the state of each robot on different cameras by their filter uncertainty. If a robot is no longer visible on a camera, its uncertainty will grow automatically and have less influence on the merge result. At some point we then just remove the tracker from the camera where it was not visible for some time. You can actually handle the ball in the same way, at least as a starting point.
The complete vision filter is in the moduli-vision module of our AI software release.

Our goalie is handled the same way as our other robots, at least concerning navigation and control. The bang-bang trajectory will get him to the target as fast as possible. Calculating the best interception point is a different story. For the goalie the goal line is a natural aid to solve that problem as a first approach.

I hope that answered some of your questions (and didn't generate too many new ones). Wish you all the best for RoboCup 2018 and looking forward to see you in Montreal!

Cheers,

Andre Ryll
Project Leader & Founder of
TIGERs Mannheim

Am 27.02.2018 um 15:50 schrieb luc.s.barreira@gmail.com:

Hello, TIGERS!

I'm a student from RoboIME, a team from IME, an engineering institute based in Brazil. We've met in RoboCup 2017 last year in Nagoya and exchanged a bit of knowledge with some of your team members. This knowledge we gathered from you and some other teams was deeply important for us to improve our project and hence become champions of Latin American Robotics Competition later last year!! Whenever we are lost about solutions in our project, your publications and software are good starting points in our researches.

Now it is one of these times. We are currently facing some problems we couldn't figure out on how to solve by just dissecting the publications and codes we've stumbled upon. We wonder if you could help us understand some of your implementations.

We are currently mainly having path planning, navigation, and filtering problems.

About path planning and navigation: as far as I know, you get a target position for the robot then compute a bang-bang trajectory profile to the target point respecting current global speed. As a result, you get a trajectory that gives your robots' position, velocity and acceleration at any given time. Then your robots receive this reference and deal with following it by using a PD controller with aid of sensor fusion algorithms. What isn't clear to me is: how is this path profile generated?
We are currently trying RRT for planning our path by generating an ordered sequence of coordinates (the path), and for navigation, we use a naive approach to steer, using a PID controller that outputs the robot velocities, the closest point until a certain distance, then hop to steer the next point and so on. All done by our central AI software.
The problem is that planning with RRT has been shown to be computationally cumbersome, and this navigation approach hasn't shown to be effective as well. We've tried some optimization suggestions from old CMDragons TDPs to no avail. And the many TDPs we've found don't explain lower level details of navigation, so we wonder if you could help us with your solutions or guide us at some expense. Another topic that could help us is how to extract information such as estimated arriving time of a given path (is assuming the bang-bang profile all along the path precise enough?), and when should the paths be recalculated, since we are dealing with 4x60fps, recalculating whenever a frame comes in seems ineffective, as well as setting a fixed time to do so (which is one of our naive solutions). Is prioritizing some robot roles - such as offensive robots - an efficient approach? How you deal impossible paths (wonder enemy robots surrounding the ball)? These are our main concerns.

About filtering: I've found in your latest publication of software that you apply an exponential filter on the cameras (I imagine there's more behind it, but couldn't figure out). We apply a Kalman filter for each camera (based on given id) and then, on redundant robot information that happens on intersections, we choose the "oldest" one seen by that camera. We feel like the "camera choosing" heuristics and Kalman filtering model could be improved, and by trying to understand how you filter the cameras' information made us curious about how it is really done. We would be glad to have an explanation for these.

Another concern is the goalkeeper controlling. We currently use the same controls for each robot, and that has shown to be underwhelming for the goalkeeper's role of interception. Is planning, navigation and control dealt differently for goalie? We are confused on how dealing with the goalie without having to cheat our software architecture and good practices. We wish to have some help with that too.

We will be competing for division B in the RoboCup 2018, and we are thirsty for becoming this year's B champions! We fondly thank you in advance for any support, and wish you the best results in division A!

Greetings,

Luciano Barreira.