MEAM 620 Project 3A

Alex Burka, Sarah Costrell, Conor O'Brien

May 5, 2015

1 Description of Problem and Associated Algorithms

1.1 CAPT

The concurrent assignment and planning of trajectories problem, or CAPT, involves finding a method of assigning N homogeneous robots to M goals and generating collision-free paths in order to reach the goals. The linear assignment portion of this problem may be offloaded to the Hungarian Algorithm, which is of complexity order $\mathcal{O}(N^3)$. Robots are generally assumed to be point-set objects in a ball of radius R.

1.2 C-CAPT

C-CAPT is a centralized solution to the CAPT problem, via which trajectories are minimized via a cost functional encompassing valid assignment, resource utilization (with respect to the assignment matrix), initial conditions, terminal conditions, robot capabilities (the dynamics of each robot, generally assumed to be first-order), and collision avoidance. The trajectories have minimum velocity squared, which also leads to their being collision-free.

1.3 D-CAPT

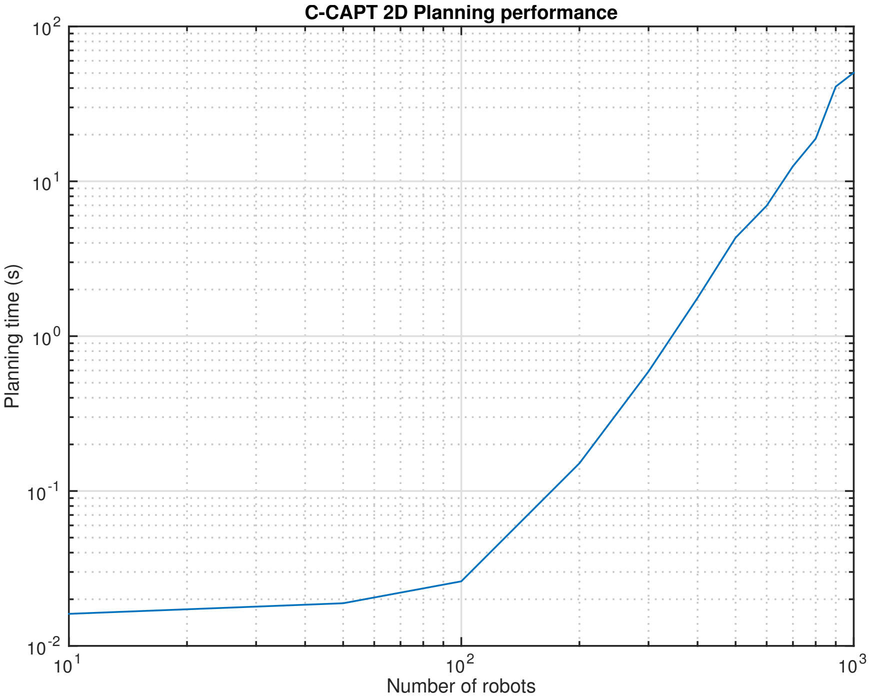
D-CAPT is a mainly decentralized solution to CAPT, using communications between robots in radius h neighborhood of one another to coordinate local actions. While this approach is directed at systems without full connectedness, the D-CAPT method helps to ameliorate issues of computational expense associated with the Hungarian Algorithm on large networks, and as such is useful in improving performance in certain centralized cases as well. [Turpin 2014] shows that this approach is successful in non-pathological examples where the number of robots is equal to the number of goals.

2 Implementation and Runtimes

2.1 C-CAPT (2D)

Our implementation of C-CAPT is included in the attached code folder. For solving the Hungarian problem, we used the toolbox available on the Mathworks File Exchange by

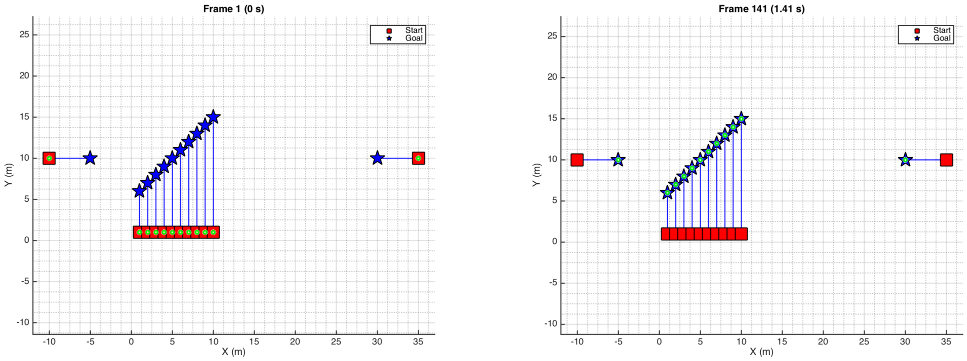
Markus Buehren (http://www.mathworks.com/matlabcentral/fileexchange/6543). To see our algorithm in action, either run demo.m in the code folder, or watch the videos included in the video folder.



We produced this graph by averaging 10 trials at each number of robots for a given set of start and goal points that satisfy the assumptions needed for C-CAPT. As we can see it is comparable to the timing found in the paper.

2.2 Interesting D-CAPT Examples

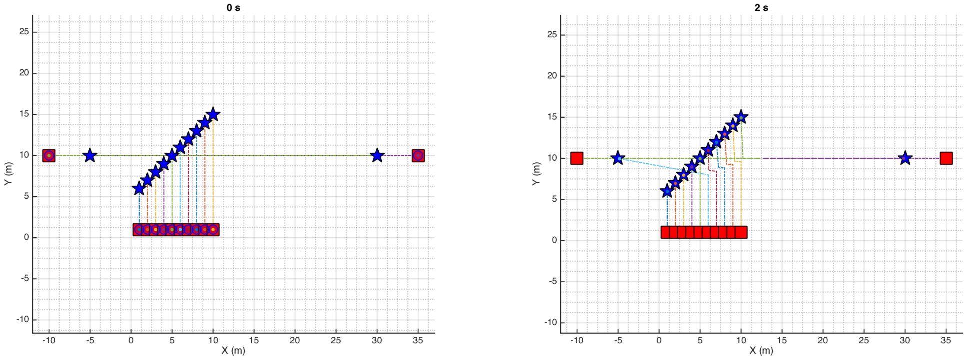
Figure 1: Example #1 Using C-CAPT



Initial State

End State

Figure 2: Example #1 Using D-CAPT



Initial State

End State

Figure 3: Example #2 Using D-CAPT

Initial State End State

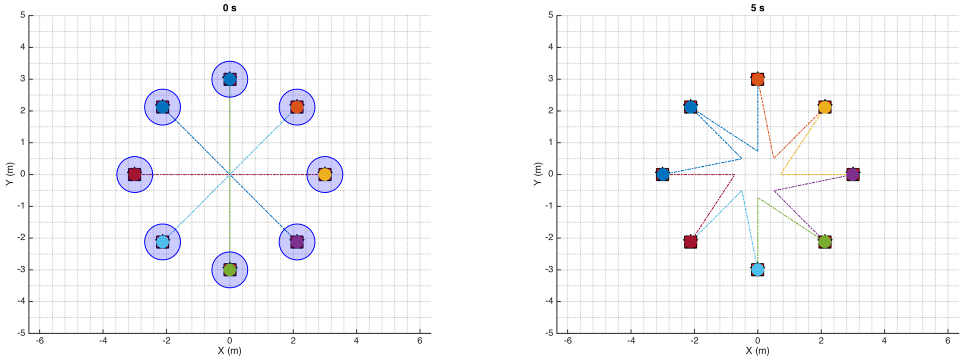


Figure 4: Example #3 Using D-CAPT

Initial State End State

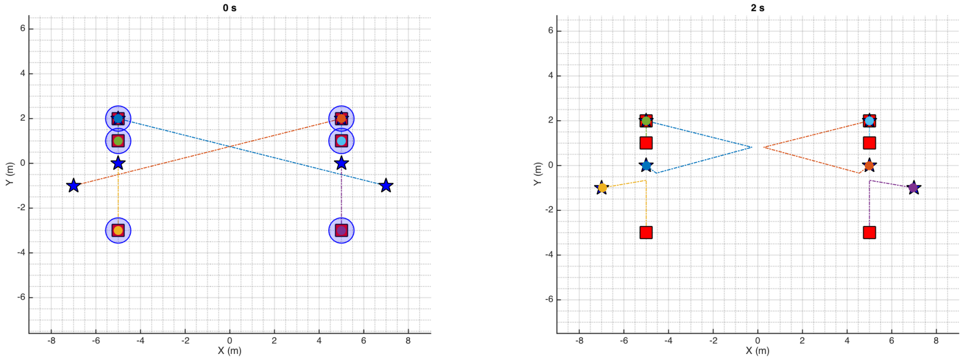
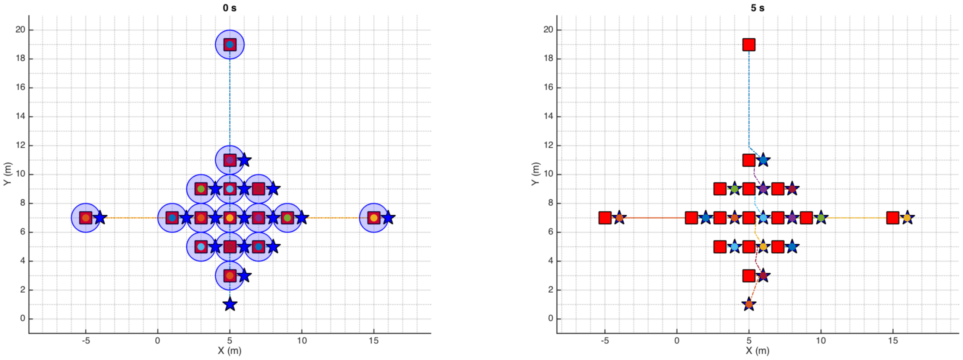


Figure 5: Example #4 Using D-CAPT

Initial State End State



These four examples show the main drawback of D-CAPT compared to C-CAPT, which is that since points are not assigned optimally at the start, there can be extremely non-optimal paths. Other drawbacks include that the number of robots must equal the goal points (however we implemented a small extension to allow this), the inital assignment problem must be unique without a centralized controller, and that networking and goal-swapping precedence creates a leyer of complexity in robot-to-robot communication.

3 Further Possible Work

It may be possible to convert D-CAPT in particular to a coordinate-free problem, e.g. by using homological methods on the simplicial complex formed by each proximity set. By working only with the abstract combinatorial object of a simplex, computational costs could be further lowered. Additionally, since the tools of homology are best-suited for the detection of holes in a network, this approach may lead to a more efficient way of selecting agents to generate proximity sets. Since homological computations are matrix-based, this may also have benefits with respect to computational costs.