Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 49-1 (2016) 142–147

Minimum Time Collision-Free Trajectories for Grabbing a Non-Tumbling Satellite ⋆

T Venkata Bhargava * **K Kurien Issac** **

* *Team Indus, Bengaluru, India (e-mail: venkat.bhargav28@gmail.com)*
** *Indian Institute of Space Science and Technology, Valiamala, Thiruvananthapuram 695547 India (e-mail: kurien@iist.ac.in)*

Abstract: The problem of grabbing a non-tumbling satellite, addressed in this paper, is split into two subproblems, the first of determining a minimum distance collision-free path from the starting stowed pose of the space manipulator to the grabbing pose, and the second of determining a minimum time motion along this path, subject to speed and torque constraints of the manipulator. The first problem is solved using the well known A* algorithm, after posing path planning as a graph search problem. Different types of heuristic functions are used for the graph search, and their influence in finding the solution is discussed. The non-smooth path generated using A* algorithm is smoothened, and then minimum time motion subject to constraints is determined by posing it as a nonlinear programming problem and solved using an SQP algorithm. Solutions obtained for three different target satellite poses are discussed.

Keywords: satellite grabbing, path planning, graph search, minimum time trajectory.

1. INTRODUCTION

Service satellites are used for repairing malfunctioning satellites and for deorbiting defunct satellites. Using a manipulator on the *chaser satellite* to grab a *target satellite* and brake its relative motion, is an important related task, and has already been accomplished by Space Shuttle with its robotic arm being directly controlled by astronauts onboard. Executing this task autonomously has been achieved successfully in two experimental missions, ETS VIII mission of 1999 (Inaba & Oda 2000), and Orbital Express mission of 2007 (Ogilvie et al 2008), in which target was first released from chaser, and then captured.

The satellite capture problem has been addressed in the past, with several approaches proposed for various aspects of the problem. Moving a manipulator without disturbing or minimally disturbing the chaser on which it is mounted, estimating the motion of a tumbling target satellite and planning a collision-free path to grab it, and grabbing the target satellite without disturbing it too much through contact, are some of the problems addressed in literature. Shiller & Dubowsky 1991 addresses the problem of determining time optimal motions of manipulators in the presence of stationary obstacles, using the *branch and bound* approach to evaluate and classify a set of identified collision-free paths, and progressively refining them. The configuration space is discretized into a rectangular grid and augmented by a direction vector to enable moves without sharp changes in direction. The final refinement to obtain the solution is based on the well known approach of Bobrow et al 1985, and Shin and McKay 1985. Lampariello 2010 defines trajectories in joint space using B-splines, uses inequality constraints to avoid collisions, and maximizes manipulability or clearance with obstacles. Though proposed for tumbling target satellites, the approach can be used for non-tumbling satellites too. Aghili 2012 also addresses the problem of capturing a tumbling satellite, and uses visual feedback processed using a Kalman filter, minimizing a composite objective function of time, distance, and line of sight angle, with constraints on accelerations. Wang et al 2015 minimizes disturbance to chaser satellite or maximizes manipulability, with constraints on joint variables and their first and second derivatives, with trajectories being represented by Bezier curves.

We follow an approach which has some aspects similar to some of the above approaches (Bhargava 2015). We assume that the chaser is stationary with respect to the target, and is maintained so even when the manipulator is moved. We also confined our problem to a planar one, but the algorithms developed are in principle applicable to the spatial case. We first find a collision-free path for the manipulator, and then find the minimum time motion along the path. In the next section we describe how the first problem is formulated and solved in a discrete configuration space. In the section after that we describe how the non-smooth path thus obtained is smoothened and parametrized for time optimization. In the section after that we formulate the minimum time trajectory determination problem and describe how it was solved. In the last section we summarize the major conclusions from the work.

2. COLLISION-FREE PATH PLANNING

We chose the dimensions of chaser and target satellites to be somewhat similar to that of the ETS VII mission (Oda 1994). The chaser, manipulator in stowed pose, and target are shown in Fig. 1, and the relevant planar dimensions are mentioned in the caption of the figure.

⋆ This paper is based on the MTech Thesis of the first author.

10.1016/j.ifacol.2016.03.043

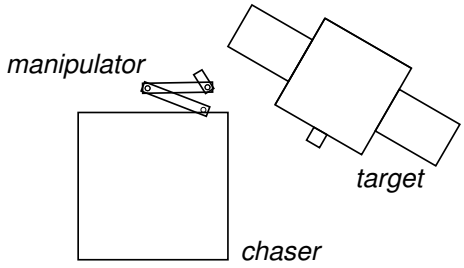

Fig. 1. Chaser (3 × 3 m), manipulator (link lengths 1.3, 1.3, 0.4 m, width 0.2 m), and target (2 × 2 m, solar panels 1.5 × 1 m). Manipulator is in its stowed pose.

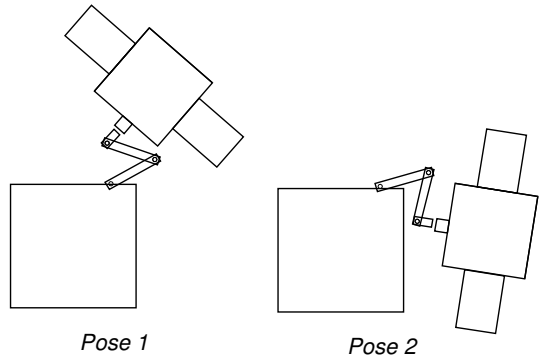

Fig. 2. Goal pose 1: $\theta_1 = 30^o$, $\theta_2 = 130^o$, $\theta_3 = -110^o$; goal pose 2: $\theta_1 = 10^o$, $\theta_2 = -120^o$, $\theta_3 = 100^o$.

2.1 Discretization and Graph Search

The three joint angles of the planar 3R manipulator are used to define its configuration space. We assume that the joint motions are restricted to $10^o \leq \theta_1 \leq 170^o$, $-160^o \leq \theta_2 \leq 160^o$, $-160^o \leq \theta_3 \leq 160^o$, to prevent collisions between adjacent links which are assumed to be in the same plane. A grid is formed in the above three dimensional parallelopiped at intervals of 10^o in all three directions. For the present study, in all cases, the start manipulator pose is taken as the stowed pose $\theta_1 = 170^o$, $\theta_2 = -160^o$, $\theta_3 = 160^o$, although it is conceivable that depending on the final relative pose of the chaser and target, it would be useful to bring the manipulator out of the stowed pose to a better starting pose, for more effective grabbing. For the present study, we selected some of the grid points as the goal poses, after checking whether the target placed with grab point appropriately positioned, is not interfering with manipulator or with chaser. A couple of such goal poses are shown in Fig. 2. It should be noted that the target pose may not correspond to a pre-specified grid point exactly. We can handle this by generating grids based on start and goal poses, or use a fixed master grid to which start and goal poses can be appended as intermediate points.

Each grid point is considered as a node of the graph. In the three dimensional grid, an inside grid point has a maximum of 6 neighbours when only rectangular moves are allowed, and has 26 neighbours, when diagonal moves are also allowed. Diagonal moves help shorten the path. Neighbouring nodes which represent collided poses of the manipulator, or cannot be reached because there are collided poses in the straight to reach it, are not connected (see Fig. 3 for the above situations in 2D). Having defined

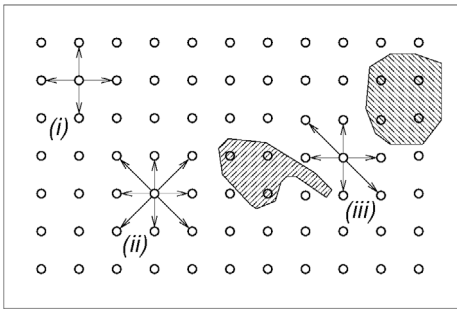

Fig. 3. Nodes and neighbours in 2D space. (i) Only rectangular moves allowed; (ii) Rectangular and diagonal moves allowed, giving 8 neighbours; (iii) neighbours reduced to 6 due to collisions.

nodes and their connectivities as above, we now have a graph.

For finding the optimal path from any given *start node* to any given *goal node* in the above graph, we use the well known A* algorithm, one of the most efficient for graph search. A* algorithm guarantees the optimum path under certain conditions on the *merit function* needed to guide the search. The merit function is defined generally as $C(i) = G(s,i) + H(i,g)$, where s is the start node, g the goal node, and i some node which was generated during the search. $G(s,i)$ is the actual cost of going from s to i, along the minimum cost path from s to i. $H(i,g)$, called the *heuristic function*, is an estimate of the minimum cost of going from i to g. Thus $C(i)$ is the estimated minimum cost of going from s to g through i.

As $G(s,i)$, we add up the cost of moving between pairs of neighbouring nodes along the path from s to i. The cost between two neighbouring nodes is chosen as the Euclidean distance between the nodes, as the moves between neighbouring nodes are along the straight lines between them. We do not assign different weights for different joints, though it is natural to assign higher weight for proximal joints of a serial chain manipulator. We now discuss the importance of the heuristic function $H(i,g)$ and describe three different functions used by us.

2.2 Heuristic Functions

It can be proven that if $H(i,g) \leq G(i,g)$, then A* will find the minimum $G(s,g)$ path from s to g (Hart et al 1968). If H is an overestimate of G, then a sub-optimal path may be declared as optimal. On the other hand, if H is too much of an underestimate, then too many nodes may get explored before the optimum path is found (Diajkstra's algorithm being the extreme case of this). Since we based G on the Euclidean distance, it is natural to choose Euclidean distance for H too, that is, $H(i,g) = ||g - i||_2$. If only rectangular moves are used (no diagonal moves), then a closer to G version of H would be $H(i,g) = ||g - i||_1$. As diagonal moves are allowed, we can have an H which is exactly equal to G in the absence of obstacles, using the following definition.

$$H(i,g) = \sum_{k=m}^{1} a_k \sqrt{k}, \tag{1}$$