Robotics and Autonomous Systems 82 (2016) 1–14

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

# Unified framework for path-planning and task-planning for autonomous robots

Pablo Muñoz *, María D. R-Moreno, David F. Barrero

*Universidad de Alcalá, Departamento de Automática Campus Universitario, Ctra. Madrid-Barcelona, Km. 33, 600, 28871 Alcalá de Henares, Spain*

## HIGHLIGHTS

- A planner for mobile robotics applications is proposed.
- Integrating task-planning and path-planning provides several advantages.
- Using specific and domain independent heuristics improves the solutions generated.

## ARTICLE INFO

*Article history:*
Received 30 January 2015
Received in revised form
4 April 2016
Accepted 22 April 2016
Available online 11 May 2016

*Keywords:*
Task-planning
Path-planning
Autonomy
Robot control architectures

## ABSTRACT

Most of the robotic systems are designed to move and perform tasks in a variety of environments. Some of these environments are controllable and well-defined, and the tasks to be performed are generally everyday ones. However, exploration missions also enclose hard constraints such as driving vehicles to many locations in a surface of several kilometres to collect and/or analyse interesting samples. Therefore, a critical aspect for the mission is to optimally (or sub-optimally) plan the path that a robot should follow while performing scientific tasks. In this paper, we present UP2TA, a new AI planner that interleaves path-planning and task-planning for mobile robotics applications. The planner is the result of integrating a modified PDDL planner with a path-planning algorithm, combining domain-independent heuristics and a domain-specific heuristic for path-planning. Then, UP2TA can exploit capabilities of both planners to generate shorter paths while performing scientific tasks in an efficient ordered way. The planner has been tested in two domains: an exploration mission consisting of pictures acquisition, and a more challenging one that includes samples delivering. Also, UP2TA has been integrated and tested in a real robotic platform for both domains.



## 1. Introduction

Nowadays, most of the robotic systems that are being designed have to move and perform tasks in a variety of environments. In order to do that, they have to avoid obstacles, find a collision free trajectory, and plan tasks. Some of these systems aim to help with ordinary, but tedious tasks. Generally, they move in known surfaces and the number of decisions is usually limited such as night surveillance tasks.

However, new science missions such as the two main missions *ExoMars* and *MARS Sample Return* (part of the Aurora program of the European Union Council of Research) will require more capable systems to achieve the goals that they are designed for. Mobility and science capabilities of the rovers are increasing, which require more powerful tools to assist on-ground operators and autonomous capabilities for the rovers. In previous missions, optimality in the path followed by the rover was not a difficult task due to the short distances they could travel per day. However, a critical aspect in future missions is to optimally (or sub-optimally) plan the path that a robot should follow while performing scientific tasks. In addition, improvements for such domains can also be incorporated into commercial robotic applications, e.g. performing inventory tasks in a large warehouse or autonomous logistics domains [1].

In this direction, the paper presents some of the results obtained within the Ph.D. program on the topic of *Cooperative Systems for Autonomous Exploration Missions* supported by ESA. Particularly, we are pursuing autonomous navigation with a rover for an exploration domain with several scientific targets along a known terrain. To face this problem, we initially had a complex deliberative model expressed in Planning Domain

* Corresponding author.
*E-mail addresses:* pmunoz@aut.uah.es (P. Muñoz), mdolores@aut.uah.es (M.D. R-Moreno), david@aut.uah.es (D.F. Barrero).

http://dx.doi.org/10.1016/j.robot.2016.04.010
0921-8890/

Definition Language (PDDL) [2] in which path-planning and task-planning were solved with a PDDL planner. This solution was very limited in (i) the map size because the planner only solved small grids in a reasonable time, and (ii) the quality of path generated because the planner generated suboptimal paths that were impossible to improve by using recent research advantages on path-planning algorithms. A second approach was to partially detach path-planning and task-planning. Before the search process starts, it is possible to include in the PDDL model a visibility graph with connections between tasks, which is generated using a path-planning algorithm. Then, the PDDL planner can obtain a feasible solution to achieve all tasks. Finally, the path-planning algorithm generates paths for the movements between each pair of locations in the plan. However, in our experiments we observed that solutions were not optimal. This was because of the domain independent heuristic employed by the task-planner. Considering a movement in the same way as other actions (e.g. take picture) leads to not properly consider the distance between tasks.

For this reason, we have developed an Artificial Intelligence (AI) planner that integrates capabilities of path-planning and task-planning. The main idea is to take advantage of path-planning heuristics and merge them with domain independent heuristics to generate better solutions in robotic domains. The proposed planner, called Unified Path-Planning and Task-Planning Architecture (UP2TA), is able to plan paths considering the shortest path while performing scientific tasks in an efficient ordered way. A PDDL planner is responsible of ordering the tasks while a path-planning algorithm searches for the route between tasks. In UP2TA, these planners are highly coupled, allowing to merge the heuristics of both planners in order to provide better solutions for mobile robotics domains. We perform an experimental evaluation of our planner on two classical exploration domains: pictures acquisition and samples delivering. In addition, UP2TA is currently deployed as the deliberative layer of the Model-Based Autonomous Controller (MOBAR) [3,4], which has been also used in our evaluation.

The next section reviews those heuristic algorithms for path-planning that are used within our planner. Section 3 presents a brief description of heuristic planners in the state-of-the-art, with a special focus on the FF planner that we use in our integration. Section 4 defines approaches that interleave task-planning and motion/path-planning. Section 5 describes our initial attempt to solve problems without a heuristic integration schema. Section 6 introduces two application domains that are then used in an experimental evaluation, which is shown in Section 8. Section 9 presents some conclusions.

## 2. Path-planning review

In this section, we review the most common path-planning algorithms and their features. In particular, we focus on heuristic search algorithms applied to path-planning since we use those in our integration.

### 2.1. Representation and notation

In classical path-planning, the environment is usually represented as a two-dimensional uniform grid with blocked and unblocked cells [5]. There are two types of representations: the node can be in the centre of the cell (centre node representation) or the node can be on the corner of the cell (corner node representation). In both cases, a valid path is the one that starts from the initial node and reaches the goal without crossing a blocked cell.

From now on, consider a node $p$ as a random node, and $s$ and $g$ as the initial and goal nodes respectively. Each node $p$ is defined by its coordinate pair $(x_p, y_p)$. A solution has the form $(p_1, p_2, \ldots, p_{n-1}, p_n)$ where $p_1 = s$ and goal $p_n = g$. For each node $p$ we require the following information:

- predecessors($p$): the predecessor node of $p$. It is mandatory that the straight line between $p$ and its predecessors does not cross blocked cells.
- successors($p$): a list of nodes that are reachable from $p$. The predecessors of each node $t$ in the list is $p$. Therefore, if predecessors($t$) $= p \Rightarrow t \in$ successors($p$).
- G($p$): the length of the path from $s$ to $p$.
- H($p$): the heuristic value of $p$, i.e., an estimation of the distance from $p$ to $g$. Typically, the A* algorithm [6] uses the Octile distance, while Theta* [7] uses the Euclidean distance.

### 2.2. Heuristic path-planning algorithms

A path-planning problem can be represented as a search tree over grids. This representation has been widely discussed. Algorithms such as A* allow us to quickly find routes at the expense of an artificial restriction of direction changes of $\pi/4$. A* Post Smoothed (A*PS) [8] tries to smooth the path obtained by A* in a post-processing step. Therefore, the resulting path may be shorter than the original, but has higher running time. If A* finds a path, the smooth process checks the line of sight between every node and the successor's successor node, removing intermediate nodes when possible. Also, there have been some improvements such as its application to non-uniform costs setting in Field D* [9], or more recently Theta* [7], which aims to remove the restriction on direction changes that A* generates. Both algorithms use the same evaluation function as in Eq. (1).

$$F(p) = G(p) + H(p). \tag{1}$$

The main difference between A* and Theta* [7] is that the former only allows that the predecessors of a node is its predecessor, while in the latter the predecessors of a node can be any node. Usually, path-planning algorithms compliant with this property (i.e. predecessors($t$) $= p \Rightarrow t \in$ successors($p$) is no longer mandatory) are called any-angle path-planning algorithms. This property allows Theta* to find shorter paths with fewer turns compared to A*. However, this improvement implies a higher computational cost due to additional operations performed during the nodes expansion process. These algorithms work on fully observable environments except Field D* that can deal with partially observable environments applying a replanning scheme.

From the Theta* algorithm, some effort has been performed in order to improve its performance such as Incremental Phi* [10] that takes into consideration the free-space assumption and angle ranges computation to provide a speed-up of approximately one order of magnitude with respect to Theta*; or Lazy Theta* [11] that delays the computation of the line of sight checking (decreasing the number of checks), and improves the performance in order to deal with 3D cubic environments.

The Smooth Theta* (S-Theta*) [12] algorithm, developed from Theta*, aims to reduce the amount of turns that the robot should perform to reach the goal. The motivation is that robots could require more time when turning, although is not desirable to rotate big angles in soft terrains. The S-Theta* algorithm considers early in the search process those nodes that follow the current heading of the robot. Then, the evaluation function of A* or Theta* is modified by adding a third parameter as in Eq. (2), which evaluates the direction of the successor node with respect to the current predecessors' position and the goal node.

$$F(p) = G(p) + H(p) + \alpha(p). \tag{2}$$

This means that, at the beginning of the search process the algorithm tries to follow the line connecting the initial node to the goal node. That is, the shortest route, if there are no obstacles. When the initial direction needs to be changed because of an obstacle in the path, the algorithm expands nodes taking