Motion Planning for Mobile Robots

Theresa Alexandra Aurelia Naß

therabyte@protonmail.com

University of Hamburg

June 19, 2019

Overview

1. Introduction
2. Configuration Space
3. Combinatorial Planning (CP)
4. Visibility Graphs
5. Vertical Cell Decomposition
6. Dynamic Window Approach (DWA)
7. Velocity Obstacles

Introduction - Motivation

Motion planning:

"[...] is eminently necessary since, by definition, a robot accomplishes tasks by moving in the real world."

- Jean-Claude Latombe (Robot Motion Planning)

- collision-free trajectories
- robot should reach the goal location on the fastest route
- reaching the goal without collisions
- encompasses several different disciplines (connected to control theory)

Introduction - Applications

Holonomic:
Holonomic refers to the relationship between controllable and total degrees of freedom of a robot.
Controllable degree of freedom is equals total degrees of freedom.

- Manipulator arms (with dynamics)

Non-Holonomic:
If the controllable degree of freedom is less than the total degrees of freedom, then it is known as non-Holonomic drive.

- Differential Drive Robots
- Moving obstacles
- Cars, Planes
- Acceleration bounded systems

Configuration Space

Quick Repetition:

Configuration

- a complete specification of the position of every point in the system

The space of all configurations is the **Configuration Space** or **C-space**.

Expressed as vector of positions and orientations.

The C-Space is obtained by sliding the robot along the edge of the obstacle regions, enlarging them by the robot's radius.

Configuration Space

Example for "circular robots"

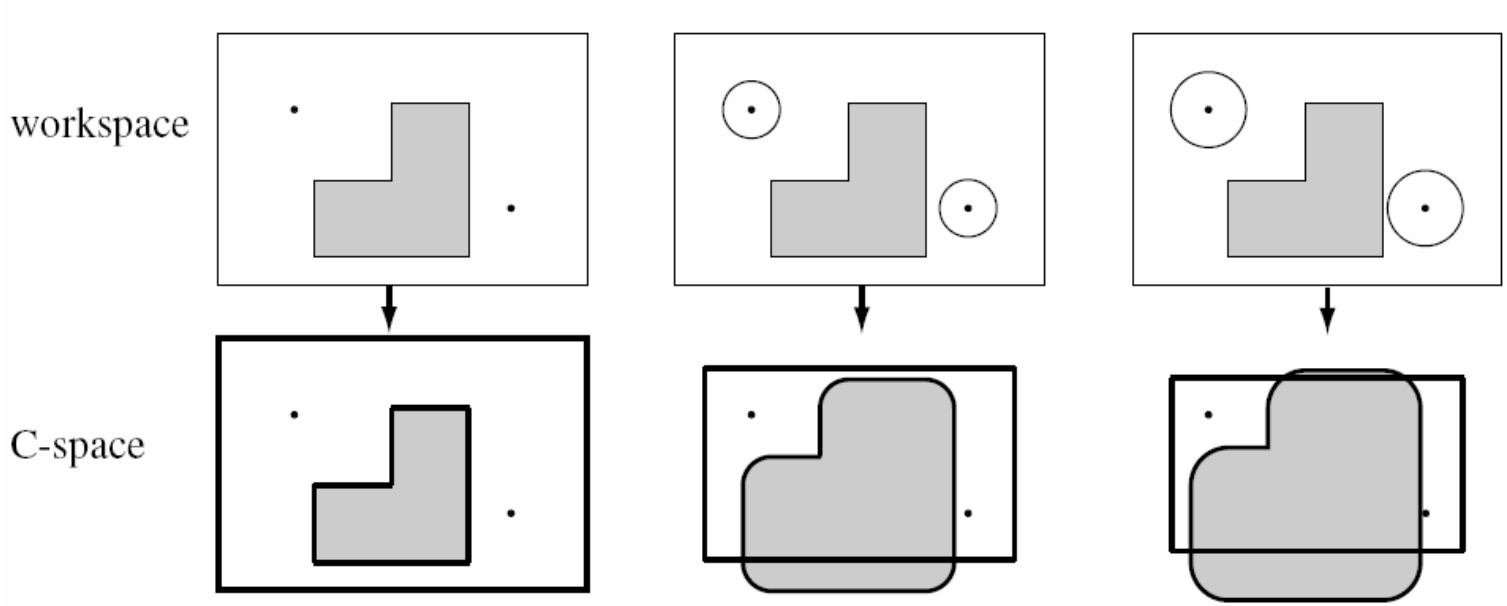


Figure: Obstacles are blown up to account for the actual size of the Robot [Robot Motion Planning, Woflram Burgard et. al]

Configuration Space

The C-Space contains **Free Space** and **Obstacle Regions**

C_{free} - set of parameters where there is no obstacles
C_{ops} - the obstacle region
Target Space - linear subspace of free space indicating where we want the robot moved to
Danger Space - space that the robot can but if possible shouldn't move to

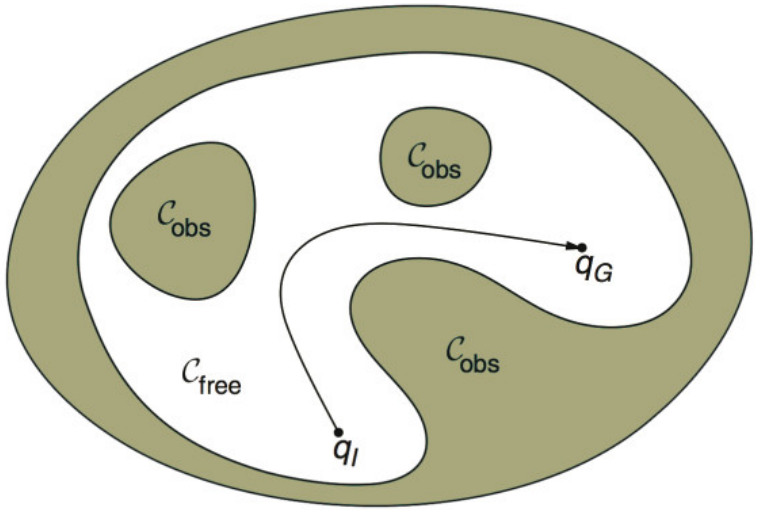


Figure: Different Spaces, Uni Freiburg, Motion Planning

Combinatorial Planning

Combinatorial Approaches

- to motion planning find paths through the continuous configuration space without approximations.

Also referred to as **exact algorithms**.

In contrast to the sampling-based motion planning algorithms.

All of them result in a **Road Map**.

These approaches are usually approaches subdivided into a two-layered architecture. (see later DWA).

Combinatorial Planning (CP)

Some **Combinatorial Planning Techniques:**

- Visibility Graphs
- Exact Cell Decomposition
- Approximate Cell Decomposition
- Voronoi Diagrams

CP methods don't scale well with C-space dimension and non- linearity but are **complete** and **optimal**

Visibility Graphs

- used to a shortest paths among a set of polygonal obstacles in the plane

One of the earliest path planning methods (late 1970s)

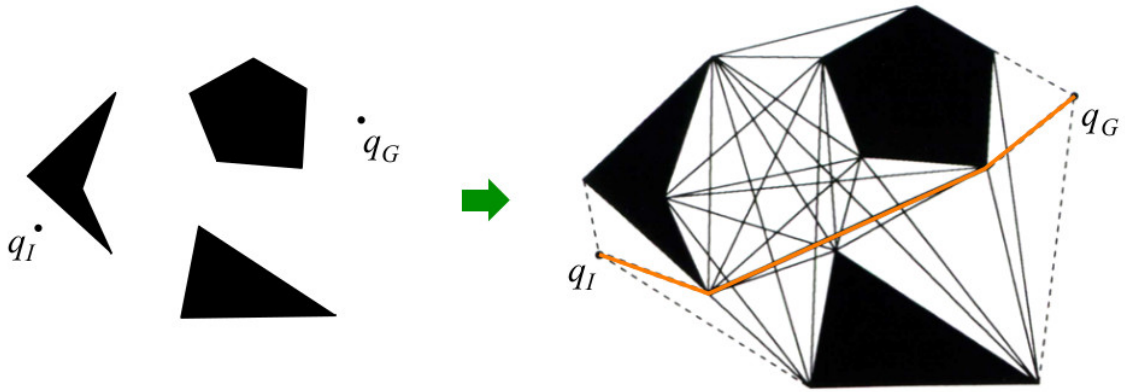


Figure: Finding a Path, Robot Motion Planning, James Kuffner et. al

Visibility Graphs

Constructing a graph between all visible vertices of the obstacles

The **shortest path** between two obstacles follows straight line segments except at the vertices of the obstacles, where the robot might turn to get around

(shortest if start and goal are at the corners of the obstacle)

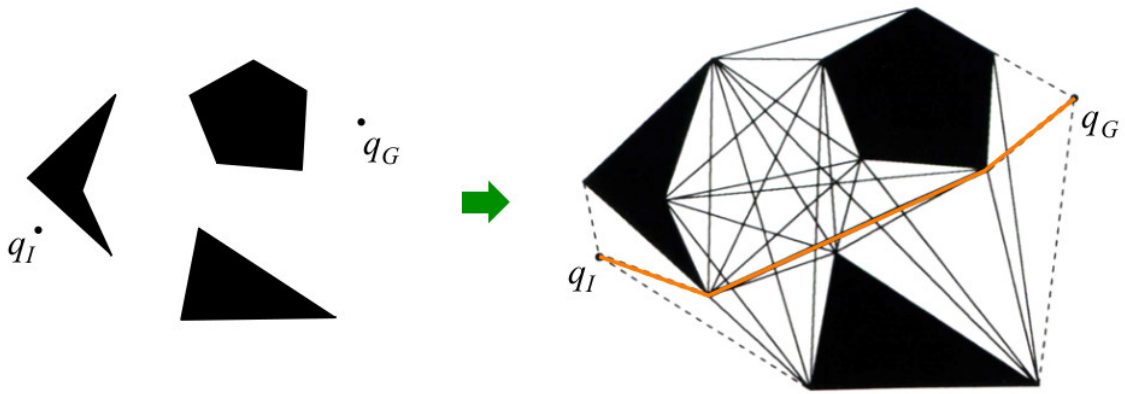


Figure: Finding a Path, Robot Motion Planning, James Kuffner et. al

Visibility Graphs

Subdivided:

- Constructing the visibility graph
- Applying a shortest path algorithm to the graph (e.g. Dijkstra's algorithm)

(a) Dijkstra's Algorithm

(b) A* Algorithm

Vertical Cell Decomposition

Vertical Cell Decomposition is an **Exact Cell Decomposition**

Can be extended to higher dimensions and non-polygonal boundaries
(cylindrical cell decomposition)

Decompose C_{free} into trapezoid or triangular cells

Planning in cells is trivial because it's convex

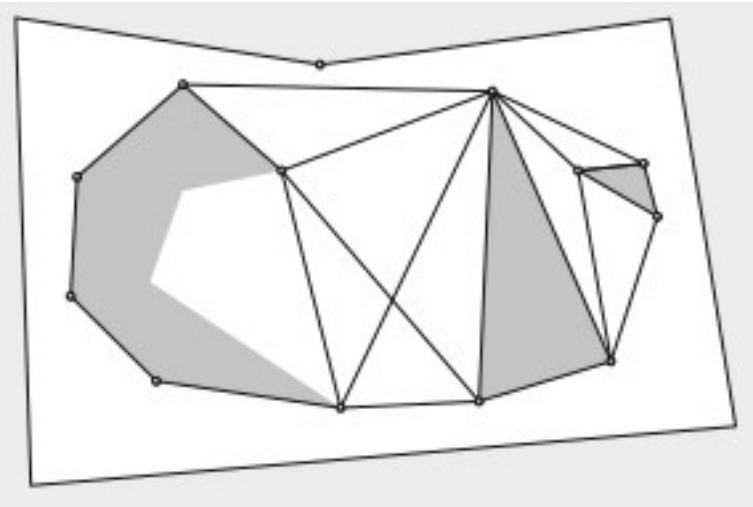


Figure: Polygonal Obstacles and Cells

Vertical Cell Decomposition

- Divide map with vertical lines
- If an obstacle would cut through, stop at obstacle

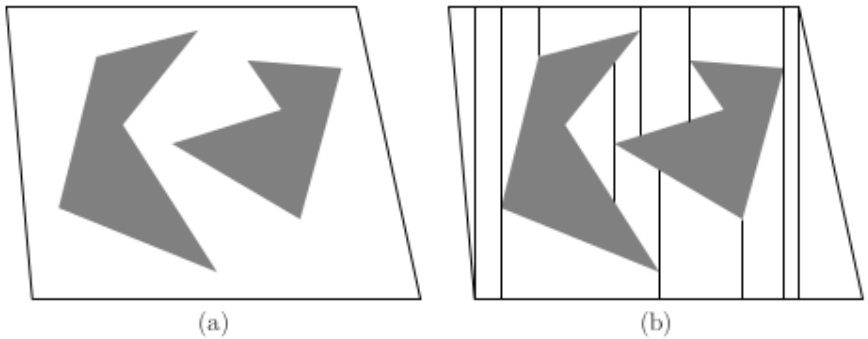
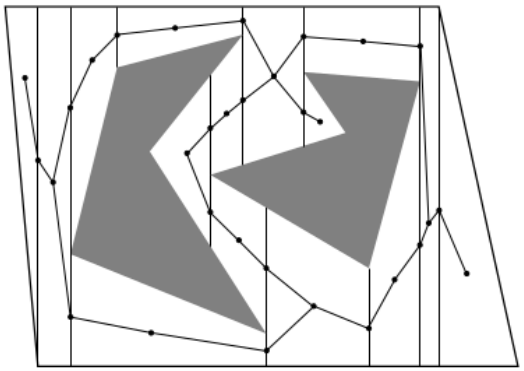


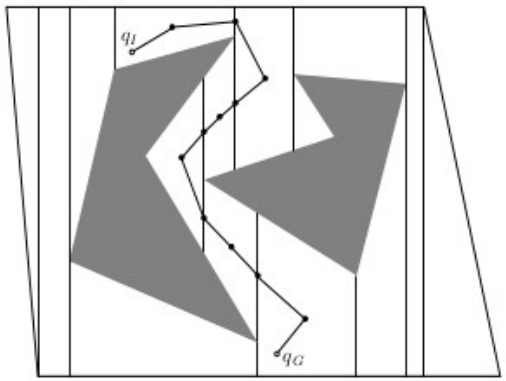
Figure: (a) Obstacles (b) Vertical Cells

Vertical Cell Decomposition

- Place one vertex in the interior of every trapezoid, pick e.g. the centroid
- Place one vertex at the midpoint of every vertical segment
- Connect the vertices to form the adjacency graph
- Use any graph search algorithm to find a collision-free path quickly



(a) Road Map



(b) Example Solution Path

Approximate Cell Decomposition

Unlike with **Visibility Graphs Cell Decomposition** doesn't get so close to the vertices

Turtle bots use occupancy grids which are similar to **Approximate Cell Decomposition**

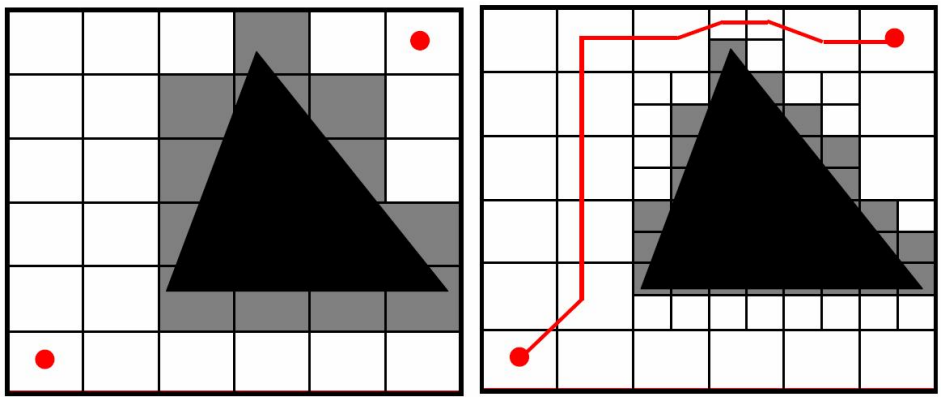


Figure: Approximate Cell Decomposition (ACD)

ACD creates a finer grid for grids that are not fully filled with the obstacle
Not an **exact algorithm**

DWA - Collision Avoidance

Collision Avoidance (Obstacle Avoidance)

well researched, different approaches exist:

- Dynamic Window Approache
- Nearness Diagram Navigation
- Vector-Field-Histograms
- Extended Potential Fields

Two types:
Methods that compute a subset of motion directions. (vector field histogram)
Methods that compute a subset of velocity controls.
(dynamic window approach and the velocity obstacles).

Dynamic Window Approach - Trajectory Planning

3 common approaches to trajectory planning:

- Potential Fields
- Sampling based Method
- Dynamic Window Approach

Challenges

- calculating the optimal path considering uncertainties in the actions
- quickly generating actions when unforeseen obstacles occur

Dynamic Window Approach (DWA)

Dynamic Window Approach:

- Simple yet effective
- Path to goal, range scan of the local vicinity, dynamic constraints
- **Goal:** collision-free, safe, and fast motion towards the goal

DWA is a compromise among **Goal**, which favors velocities that offer progress to the **goal**, **clearance**, which favors **velocities** far from the **obstacles**, and **velocity**that favors high speeds.

Dynamic Window Approach

Subdivide Motion Planning: global and local motion planning

- Approximate **global planner** computes paths without kinematics and dynamic vehicle constraints
- Accurate **local planner** with kinematics, searching an appropriate velocity space

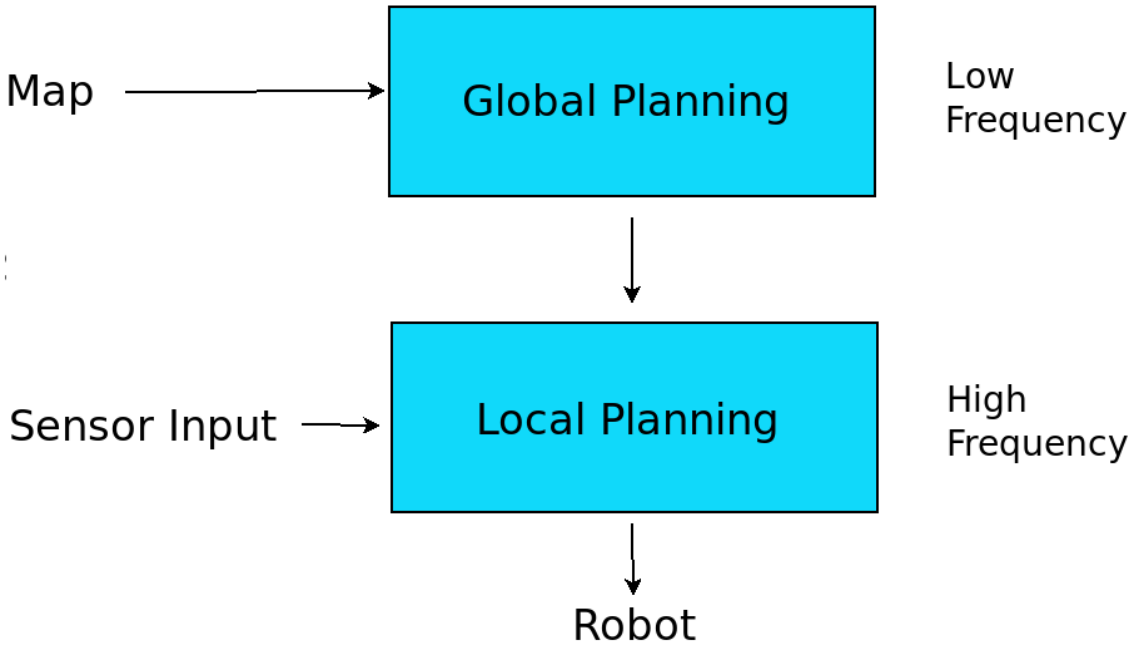


Figure: Subdivision of DWA

Global Planning

Global Planning:

Considers the entire map

Adds NF1 (navigation algorithm) or similar to the **Objective Function**

For more efficiency NF1 only calculates within a rectangular region, directed from the robot to the goal

Region will be enlarged and recalculated if the **goal** can't be reached within the chosen region

Local Planning:

Now kinematics are taken into account by searching a **Velocity Space**

Velocity Space contains all set of possible tuples (ν ω)
(ν-velocity, ω-anuglar velocity)

Assumed that the robot moves instantaneosly on circular arcs

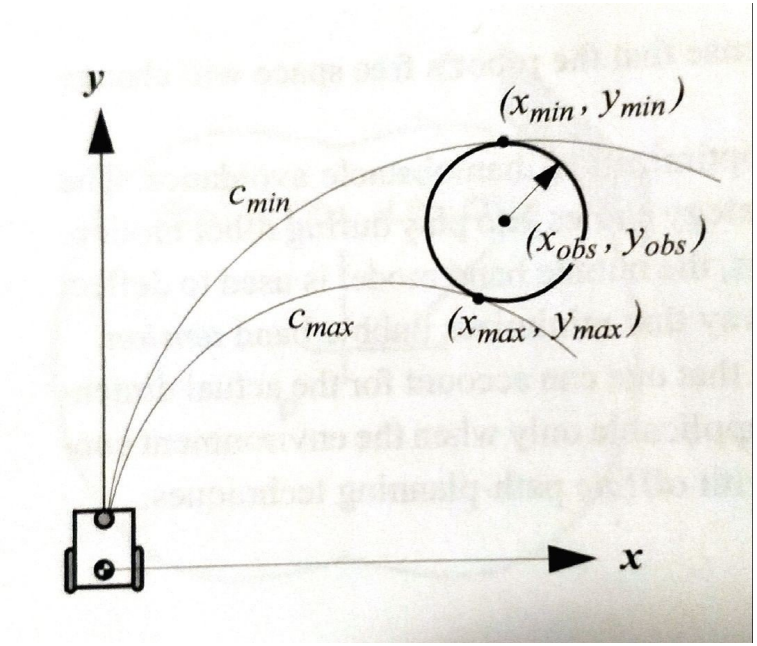


Figure: Circular Arc

DWA

Select **dynamic window** of all reachable tuples $(\nu\ \omega)$ within the next sample period dependings on current robot speed

Then choosing only tuples in which the robot can stop before collision (Admissable velocities (av))

Rectangular shape of the window is due to the approximation that the dynamic capabilities for translation and ration are independent

Dynamic Window Approach (DWA)

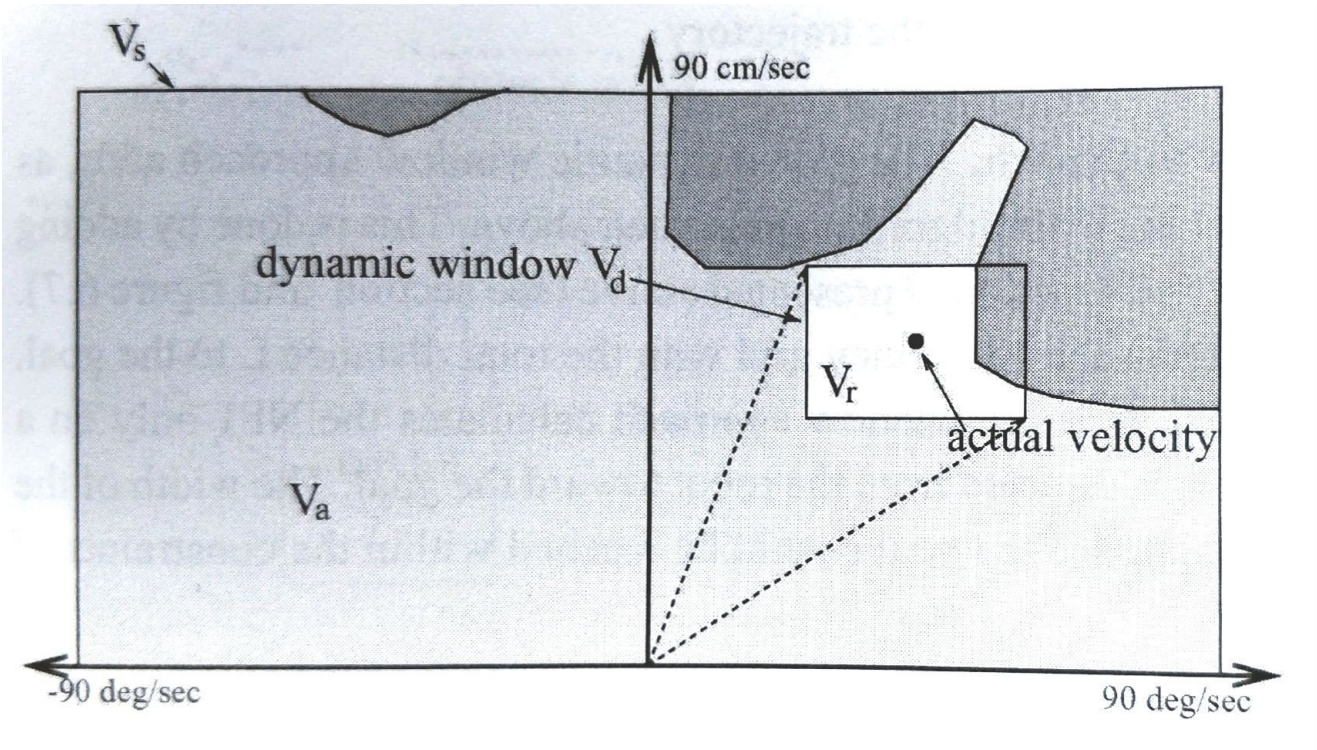


Figure: Dynamic Window Approach, Autonomous Mobile Robots, S.403

V_s = all possible speeds of the robot, V_a = free space, $V_r = V_d$ = speeds reachable in one time frame with accerlation constraints

DWA

New motion directive is chosen by applying an objective function to all *av* tuples in the DW

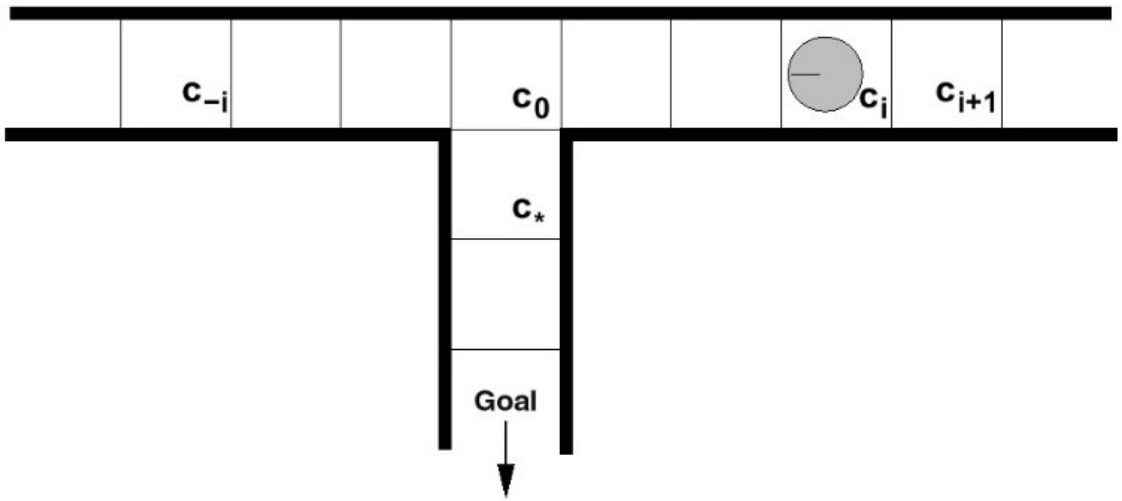
Objective Function prefers:

- fast forward motion
- maintanance of large distances to any obstacles
- alignment to the goal heading

The **Objective Function** is a heuristic navigation function
encodes the desire to minimize *driving fast, safe, in the right direction*

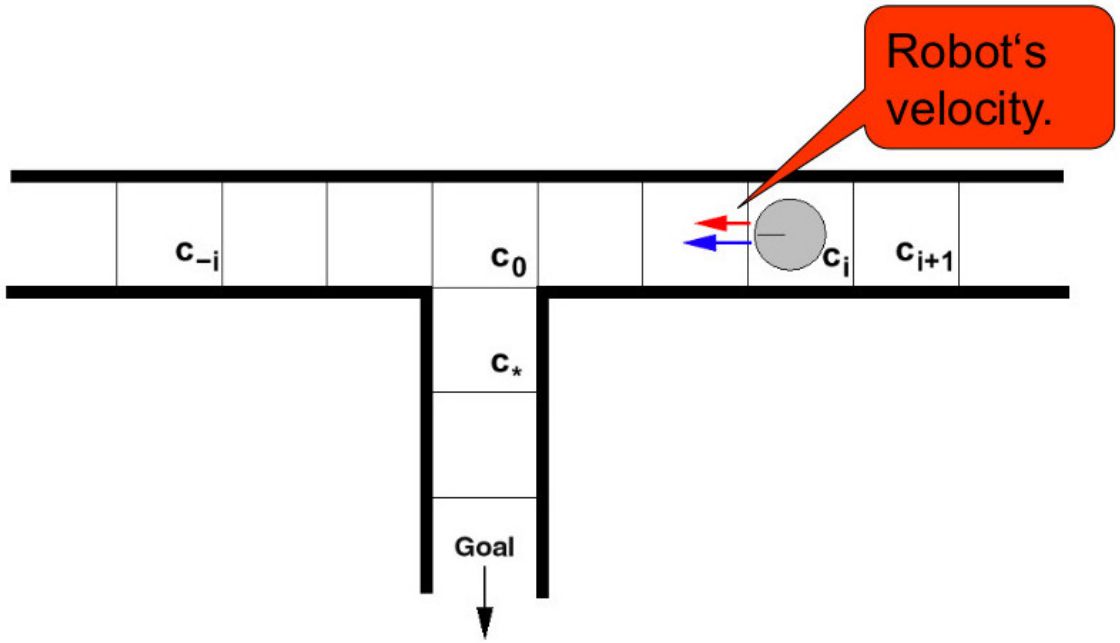
heading - measure of progress toward the goal location
velocity - forward velocity of the robot
dist - distance to the closest obstacle

DWA - Problems



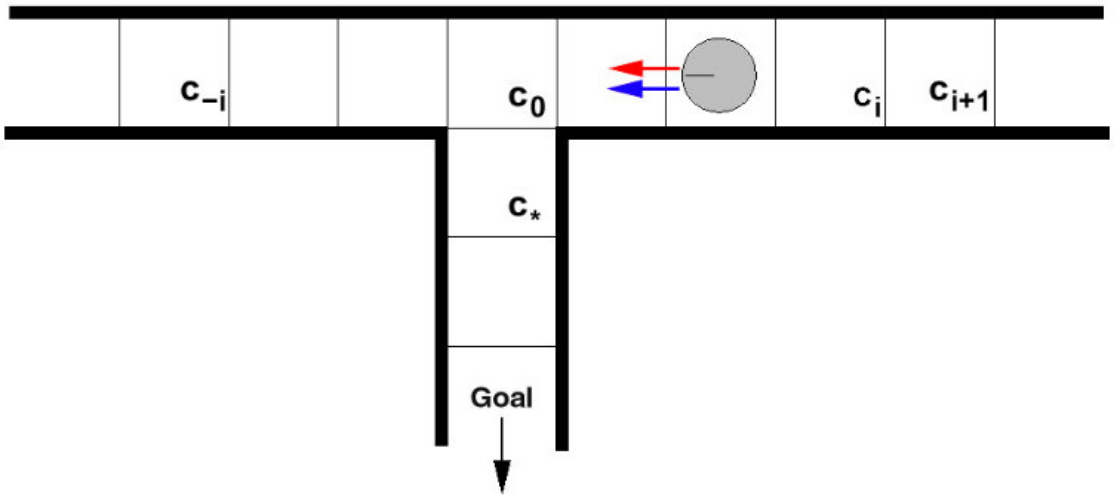
All graphics from Robot Motion Planning, Wolfram Burgard et. al, Uni Freiburg

DWA - Problems



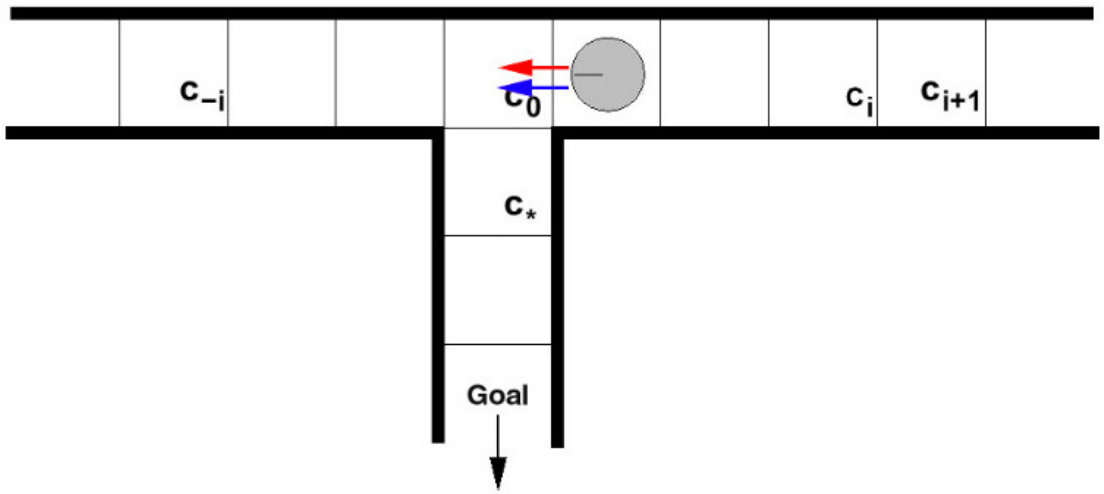
All graphics from Robot Motion Planning, Wolfram Burgard et. al, Uni Freiburg

DWA -Problems



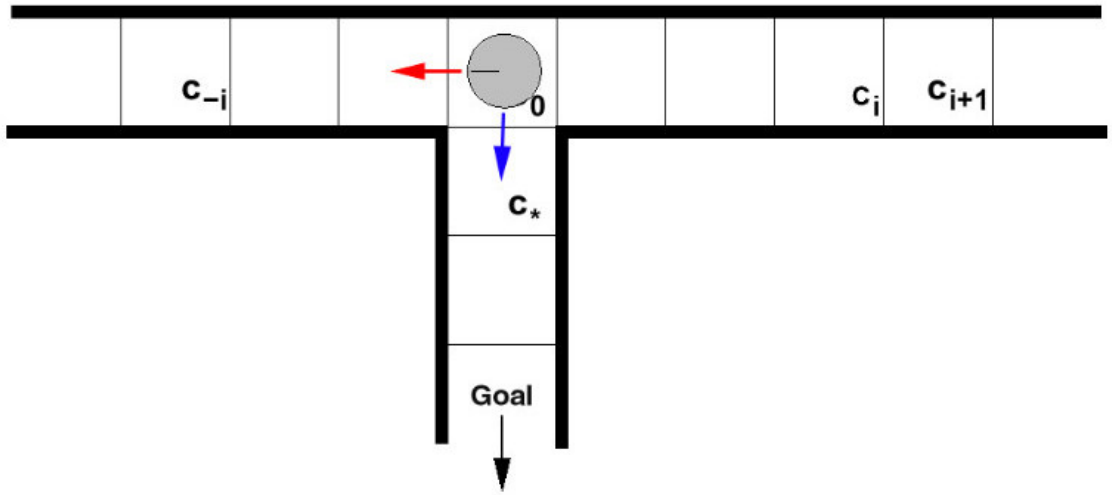
All graphics from Robot Motion Planning, Wolfram Burgard et. al, Uni Freiburg

DWA - Problems



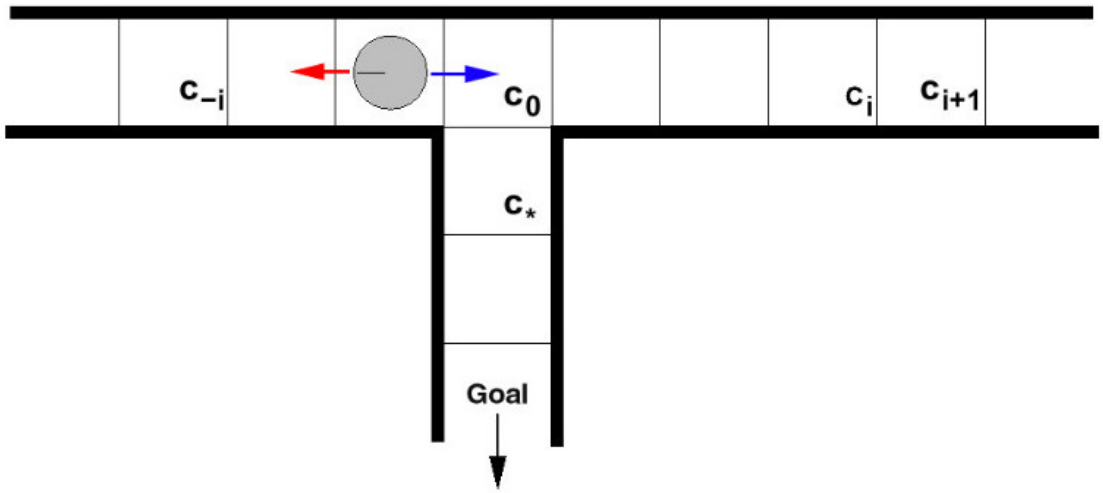
All graphics from Robot Motion Planning, Wolfram Burgard et. al, Uni Freiburg

DWA - Problems



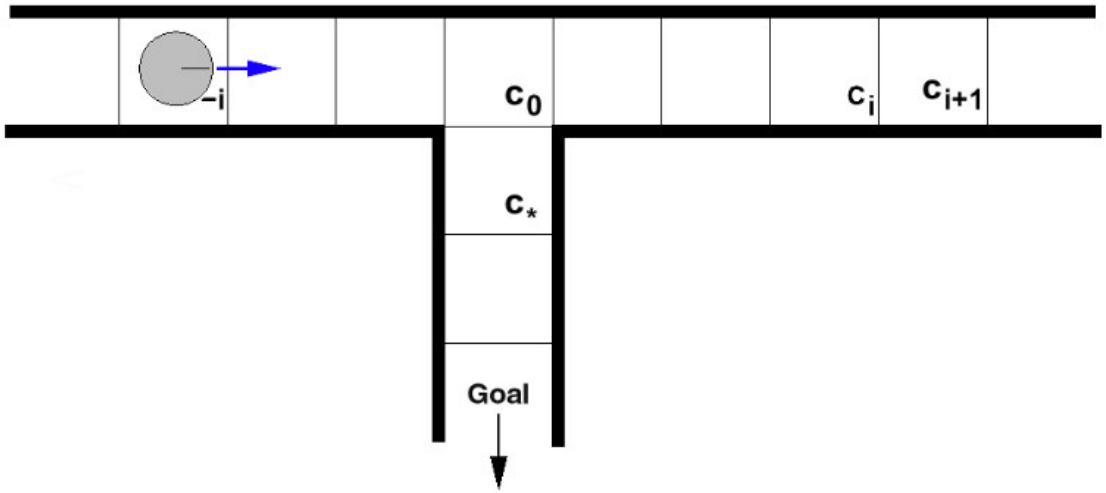
All graphics from Robot Motion Planning, Wolfram Burgard et. al, Uni Freiburg

DAW - Problems



All graphics from Robot Motion Planning, Wolfram Burgard et. al, Uni Freiburg

DWA - Problems



All graphics from Robot Motion Planning, Wolfram Burgard et. al, Uni Freiburg

DWA - Problems

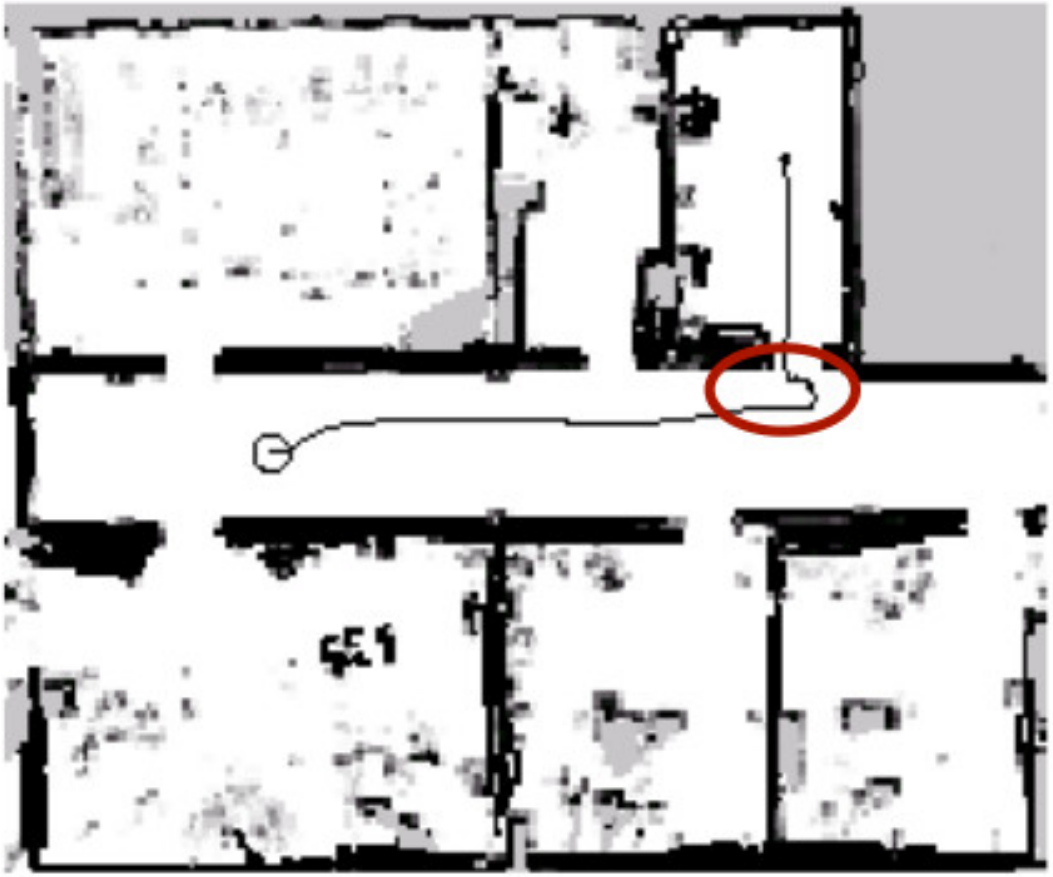


Figure: Problem in Real Life on Map, Robot Motion Planning, Wolfram Burgard et. al, Uni Freiburg

DWA - Up-/Downsides

Upsides

- Reacts quickly with low CPU requirements
- Guides the robot on a collision free path
- Successfully applied in many real-world scenarios

Downsides

- Resulting trajectories are sometimes suboptimal
- Local minima might stop the robot from successfully reaching the goal location (regular DWA)
- (A Global DWA with NF1 (a navigation algorithm) can overcome this problem)

Velocity Obstacles

Velocity Obstacles

- set of all velocities of a robot that will result collision with another robot at a point in time, assuming the other robot maintains it's current velocity.

Framework is the same as for **DWA** Method

DWA takes into account that **moving obstacles** can occur

Set of colliding trajetories is calles the **collision cone**

Set of unsafe trajectories is the union of the velocity obstacles for each moving obstacle

Takes into account the velocity of the obstacles $\rightarrow$ well suited to dynamic scenarios.

Velocity Obstacles

- Collision avoidance algorithm
- VO assumes planar motions
- Robot chooses a velocity inside the **Velocity Obstacle** the 2 robots eventually collide
- Outside the VO, a collision is guaranteed not to occur for them

Velocity Obstacles

Collision Avoidance under Bounded Localization Uncertainty (CALU)

Multiple robots navigate efficiently while avoiding each other

The robots broadcast their positions

Selecting collision free velocities which continue to adapt as the robot is reaching it's goal

CALU allows them to react to nearby robots

Doesn't require any centralized or coordinated planning

Velocity Obstacles

Collision Avoidance under Bounded Localization Uncertainty (CALU)

Approach based on **Velocity Obstacle Approach** incorporating the motion of the other robots

Leads to more efficient and smoother paths

Reactive and fully distributed approach running as local planner on the robots

Wide range of applications from warehouse deliveries to games

Velocity Obstacles

Figure: Collision Avoindance using Velocity Obstacles

start

Figure: Avoiding each other: Turtlebots and PR2

The End

and they moved happily ever after

References

Jean-Claude Latombe (1991), Robot Motion Planning

Springer Handbook of Robotics, Siciliano, Bruno, Khatib, Oussama (Eds.) 2008

The Dynamic Window Approach to Collision Avoidance, Dieter Foxy, Wolfram Burgardy, Sebastian Thrunyzy

Combinatorial Motion Planning, Steven M. LaValle, University of Illinois, 2006

Computational Geometry: Algorithms and Applications, de Berg et al., 1997

http : *//www.robotplatform.com/knowledge/Classification*$_o$*fRobots/Holonomic*$_a$*ndNon* − *Holonomic*$_d$*rive.html* (visited 29.05.2019)

Figure: Pac-Man using Velocity Obstacle Approach