# Point-point planner

## Algorithm from ROS sbpl package (search based planning lattice) algorithm

### developed by Maxim Likhachev at the University of Pennsylvania in collaboration with Willow Garage

### <http://www.cs.cmu.edu/~maxim/files/tutorials/robschooltutorial_oct10.pdf>

## Search-based ARA\* planning in 3D (x, y, theta) space

### Library of motion primitives

#### Arcs, lines, turn-in-place

#### Kinematically plausible motions

#### Cost can be assigned to each motion – e.g. make spins and reverses more expensive because they require coming to a halt

#### Natural to pick motions corresponding to path segments

### Fast

#### Nominal runtime .1-.2 sec for planning several meters in relatively clear setting

#### Runtime increases for difficult moves, especially requiring backwards motion or tight squeezes

### Kinematically plausible paths

#### Checks every pose for collision against a 2D costmap, resolution [X] cm^2, against the robot’s full footprint

#### Some pictures showing loops, back up and turn, etc

## Modifications

### Custom motion primitives for Harlie

### Changed output to produce path segments

### Correct for discretization error

# Overall planning

## Traditional point-point planning does not work for person tracking

### Whenever person moves, goal moves

#### Normally requires robot to come to halt for replanning

### Previous experiments at CWRU confirm this

#### Moving target results in stuttering

## Designed to allow dynamic replanning without the robot coming to a halt

## Splits path into two sections

### Committed path

#### Short-term plan

#### About 1m, long enough to keep the robot moving for 1-2 seconds

#### Planner tries hardest not to change this - cannot be changed without bringing robot to a halt

#### This is actually what steering sees

### Uncommitted path

#### This is what I think I’m going to do, but it could change

#### Long-term plan

#### Subject to change

## Planner monitors committed path length

### Tries to keep it around 1m (configurable

### Drops below threshold: shifts segments from uncommitted to committed if available

### If for some reason the committed path runs out (planner gets hung up because planning takes a very long time) robot comes to a halt at end of committed path to wait for more segments

## When planning gets a new goal, it plans from the end of committed path (partial replan)

### If this fails, bring the robot to a halt and replan from halt pose (full replan)

### Same happens if an obstacle is detected in uncommitted section

## Modifying committed path is done as last resort

### Obstacle detected close to robot (emergency reflex)

### Coming to a halt would be less painful – if target moves behind the robot

### Partial replan fails

# Goal generation

## Constellation

### Cannot plan right to target – space is occupied

### Generates plausible goal poses offset by angles and distances to target

### Goals in collision are eliminated

### If doing full replan, all goals are kept. If partial replan, only first 4 cleared goals are kept to keep planning time reasonable

## Special short-distance goal generation

### If target is < Xm away from start position, manually generate a spin segment

### Just turns and faces target

# Future work: separate short-distance planning algorithm

## Robot shows weakness tracking user at short range, turning around