Reactive Collision Avoidance in Shared Control for Assistive Navigation

Mark Zolotas

Personal Robotics Lab, Imperial College London

13/06/2018

Agenda

- Motion planning and reactive **collision avoidance**
- How this relates to **shared control**
- *Admissibile Gap (AG)* method [1], from the GET-Lab, University of Paderborn, Germany
- **Motivation:** To improve assistive navigation on the "smart" wheelchairs
- Video results!

What is Motion Planning?

- A field concerned with ensuring safe navigation
- **Objective:** To compute a *collision-free admissible* path to the target configuration of a mobile robot
- General motion control divided into **global** and **local** perspectives

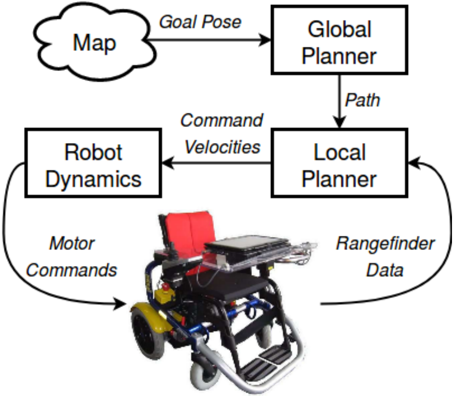

Figure 1: Generic motion control framework.

Reactive Collision Avoidance

- Same objective as motion planning but relaxes the "perfect model" assumption by taking a **sensor-based** approach
- Move towards target location safely whilst collecting information on obstacles **during** motion execution

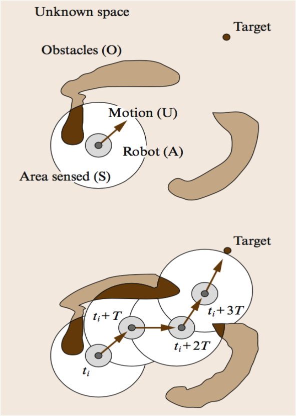

Figure 2: The obstacle avoidance problem (taken from [2]).

Typical Approaches to Obstacle Avoidance

- *Vector Field Histogram (VFH)* – Compute a range of steering directions and select the 'optimal' one
- *Dynamic Window Approach (DWA)* – Select from a set of candidate control velocities the pair (v^*, ω^*) that optimise:

$$(v^*, \omega^*) = \underset{(v,\omega)}{\operatorname{argmax}} \Big(\alpha \cdot \mathit{heading}(v,\omega) + \beta \cdot \mathit{dist}(v,\omega) + \gamma \cdot \mathit{vel}(v,\omega)\Big) \tag{1}$$

- **Gap-based** – Extract open spaces from ranger data that the robot can traverse through

Why Gap Navigation?

- Most obstacle avoidance methods struggle in dense, complex and cluttered environments
- Classic problems:
  - (i) Local trap situations
  - (ii) Oscillatory behaviour
- Gap-based methods (e.g. Nearness Diagram (ND) and its variants [3]) acquire **high-level** sensory information
- *Common limitation* across all approaches is to assume **holonomic** constraints and **disc-shaped** robots

Admissible Gap (AG) Navigation

- Directly considers the exact shape and kinematic constraints
- **Admissible Gap:** A "gap" with a single motion control that safely guides the robot through, whilst obeying the vehicle's constraints [1]
- Only 2 parameters for tuning:
  (i) d_{safe}: How far to stay away from obstacles
  (ii) D_{vs}: How to limit speed near obstacles

Algorithm

Algorithm 1 AG control loop.

Input: (x, y) - Target configuration; S - Scanner readings;
Output: (v^*, ω^*) - Assisted output command

function FindAssistiveCommand(x, y)
    if IsGoalNavigable(x, y) **then**
        $(v^*, \omega^*) \leftarrow$ GenerateMotionCommands(x, y)
    else
        $g \leftarrow$ FindClosestGap(G)
        if IsGapAdmissible(g) **then**
            $(v^*, \omega^*) \leftarrow$ GenerateMotionCommands$(g.x, g.y)$
        else
            $(v^*, \omega^*) \leftarrow (0, 0)$
        end if
    end if
end function

function UpdateObstacleMap(S)
    $O \leftarrow$ ComputeObstacles(S)
    $G \leftarrow$ ExtractGaps(O)
end function

Gap Extraction

- Evaluate rangefinder data for **depth discontinuities**
- A depth discontinuity occurs between two contiguous scan points s_i and s_j when either:

$$||s_i - s_j|| > w_{min} \tag{2}$$

  Or their ranges r_i and r_j fulfill:

$$(r_i = r_{max}) \oplus (r_j = r_{max}) \tag{3}$$

- Perform a clockwise and counterclockwise search over a full field of view scan
- **Gaps** are the resulting depth discontinuities, which are *visible* and neither *improper* nor *useless*

Checking Gap Admissibility

- Select closest gap (e.g. g_1) based on Euclidean distance from goal $\mathrm{p_g}$
- Project trajectory to a sub-goal p_s and check footprint for collisions
- Construct "virtual" admissible gaps (e.g. g_1^* for g_1) if "closest" is non-admissible

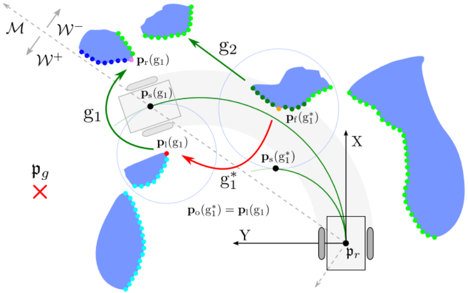

Figure 3: Gap navigability check (taken from [1]).

Generating Motion Commands

- Assume a differential-drive robot, with the kinematic constraint:

$$-\dot{x}\sin\theta + \dot{y}\cos\theta = 0 \tag{4}$$

  Where (x, y, θ) represents the robot configuration
- Trajectories of this kinematic model approximated as a sequence of circular arcs
- Generate linear and angular velocities (v^*, ω^*) according to the AG's arc tangent

Adapting into Shared Control

- Replacing the global planner for an **intention predictor**
- A global pose (i.e. the **intent**) is estimated by a forward projection of the robot's dynamics
- Find "closest" gap based on angular disparity

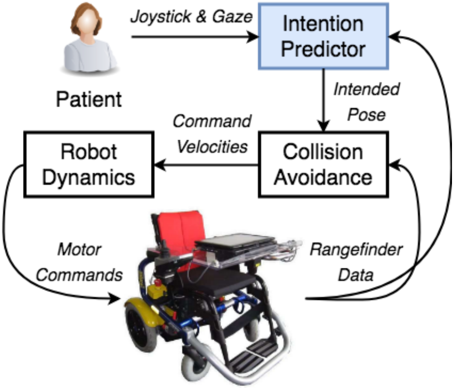

Figure 4: Shared control framework.

Challenges & Future Work

- Arcs are short due to simulated trajectories being very local; affects smoothness of motion
- Purely rotational motion not considered in the trajectory safety checks as intended pose appears static
- Need to adapt the shared controller to accommodate a global pose outside of the range limits

ARTA in Simulation

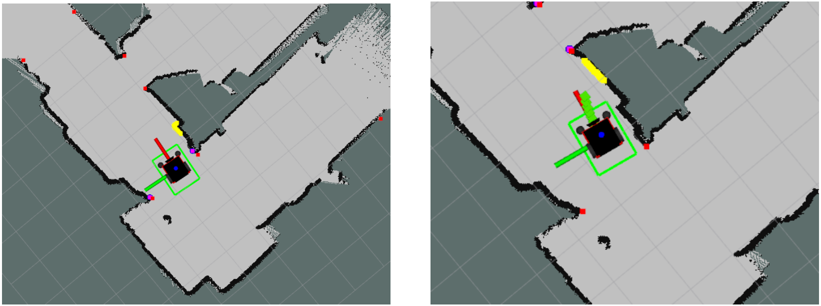

Figure 5: Gaps highlighted by red points, potential collisions are the yellow clusters, whilst the “closest” AG is the blue gap.

Thank you for listening!

References I

- M. Mujahed, D. Fischer, and B. Mertsching, "Admissible gap navigation: A new collision avoidance approach," *Robotics and Autonomous Systems*, vol. 103, pp. 93–110, may 2018.
- J. Minguez, F. Lamiraux, and J.-P. Laumond, "Motion Planning and Obstacle Avoidance," in *Springer Handbook of Robotics* (B. Siciliano and O. Khatib, eds.), pp. 1177–1202, Cham: Springer International Publishing, 2016.
- J. W. Durham and F. Bullo, "Smooth Nearness-Diagram Navigation," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 690–695, sep 2008.