# Method of Local Path Planning for an Autonomous Mobile Robot Using a Behavioural-Based Extension to the Driving with Tentacles Algorithm

Adam Panzica
Masters Candidate
Robotics and Intelligent Vehicle Research Laboratory
Worcester Polytechnic Institute
Worcester, MA 01609, USA
apanzica@wpi.edu

Taskin Padir
Title
Worcester Polytechnic Institute
Worcester, MA 01609, USA
tpadir@wpi.edu

***Abstract*— The abstract**

## I. INTRODUCTION

[1]
[2]
[3]
[4]
[5]

## II. ALGORITHM

### A. Introduction to Driving with Tentacles

'Driving with Tentacles' is a very simple path-planning system proposed by von Hundelshausen et. al. originally utilized in their DARPA Urban Challenge robot. It is a method in which a series of arcs, referred to as tentacles due to their resemblance to such when visualized, are extended outwards from the front of the robot. Each tentacle has an associated radius and velocity, and are grouped by velocity into units called 'Speed Sets.' The tentacles in faster Speed Sets are grouped more closely together with less radius of curvature but a longer arc-length, and those in slower Speed Sets further apart with more radius of curvature but less arc-length, as seen in figure **??**.

The equations for generating the tentacles, from von Hundelshausen et. al.:

$$r_k = \begin{cases} \rho^k \cdot R_j & k = 0,\ldots,\frac{k_{max}}{2}-1 \\ \infty & k = \frac{k_{max}}{2} \\ -\rho^{k-\frac{k_{max}}{2}} \cdot R_j & k = \frac{k_{max}}{2}+1,\ldots,k_{max} \end{cases} \quad (1)$$

Where $r_k =$ Radius of the kth Tentacle in the jth Speed Set, $\rho = 1.15$, and $R_j =$ Seed Radius of the jth Speed Set:

$$R_j = \frac{l_j}{\Delta\phi(1-q^{0.9})} \quad (2)$$

Where $l_j$ = The Seed Arc Length of the jth Speed Set, $\Delta\phi =$ The angle stubended by the most curved tentacle, and $q =$ a magic constant based on the number of Speed Sets, $n$:

$$l_j = l_{min} + l_{gf} \cdot q_j^{1.2} \quad (3)$$

$$q_j = \frac{j}{n-1} \quad (4)$$

$$\Delta\phi = 1.2(^{\pi}/_{2}) \quad (5)$$

Finally, the arc-length of the kth tentacle, $l_k$, is defined as

$$l_k = \begin{cases} l_j + l_{tf}\sqrt{\frac{k}{\frac{k_{max}}{2}}} & k = 0,\ldots,\frac{k_{max}}{2}-1 \\ l_j + l_{tf}\sqrt{\frac{k-\frac{k_{max}}{2}}{\frac{k_{max}}{2}}} & k = \frac{k_{max}}{2},\ldots,k_{max} \end{cases} \quad (6)$$

and the associated velocity for all tentacles of the jth Speed Set is defined as:

$$\upsilon_j = \upsilon_s + q_j^{1.2}(\upsilon_e - \upsilon_s) \quad (7)$$

where $\upsilon_s =$ minimum velocity and $\upsilon_e =$ maximum velocity of the robot.

In von Hundelshausen et. al. tentacle selection is performed by simply determining which tentacle is the longest before running into an obstacle given a radius of safety for the robot. No consideration is given to any other parameters.

### B. Extended Driving with Tentacles Algorithm

In order to improve on some of the limitations of the original Driving with Tentacles algorithm, the robot implements a modified version of the algorithm which takes into consideration a number of simple behavioral considerations useful for a robot which is performing a search-and-return mission. These include:

- Move towards some predetermined goal destination
- Move towards unexplored terrain
- Move away from previously explored terrain
- Move away from 'difficult' terrain

The desired end behavior can be described as an algorithm which will select the longest tentacle which moves the robot closest to the goal (if any) while passing through the least amount of preciously explored terrain and the least amount of difficult terrain and exploring the most new area.

To accomplish this, the local planner is given an 'Occupancy Grid' which is generated many times per second by the Global Planner. This Occupancy Grid contains a set of discretized points corresponding to a location relative to the robot-frame. Each point has a 'Point Trait', which represents

the kind of terrain at that location. The Point Trait can consist of the following types:

UNKNOWN If there is no information as to what is at this location
OBSTACLE If there is an impassible obstacle
TRAVERSED If the point has been previously explored
FREE_HIGH_COST If the point is passable but contains difficult terrain
FREE_LOW_COST If the point is open terrain
GOAL If the point is the Goal Point

All of the tentacle data is pre-generated following the equations in section II-A and discretized to the same grid parameters as the occupancy grid. As the tentacles are the same every time relative to the robot-frame, they only need to be produced once and can then be reused indefinitely.

For each new Occupancy Grid, the Local Planner iterates across each point on each tentacle in a Speed Set and examines the corresponding point in the Occupancy Grid's Point Trait. There are two values that are updated at each point $\mathbf{p}_{[n]}$ on tentacle k based on the Point Trait in the occupancy grid: The distance along the tentacle that has been traversed, which is described in (9), and the 'length modifier', which is described in (10). In addition, if the Goal Point or an Obstacle is hit, the iteration across that tentacle is immediately halted, and in the case of hitting a goal point that tentacle is selected as the 'best.'

$$\Delta l[n] = \|\mathbf{p}[n], \mathbf{p}[n-1]\|_2 \tag{8}$$

$$ll[n] = \Delta l + ll[n-1] \tag{9}$$

$$lm[n] = lm[n-1] - \Delta l[n] \cdot \begin{cases} DW & \text{if FREE\_HIGH\_COST} \\ TW & \text{if TRAVERSED} \\ -UW & \text{if UNKOWN} \\ 0 & \text{if FREE\_LOW\_COST} \end{cases} \tag{10}$$

Where $DW, TW,$ and $UW$ are weighting parameters for biasing the behaviour due their respective Point Trait

$$lg = -GoalWeight \cdot \begin{cases} 0 & \text{If no goal} \\ \|\mathbf{p}_{[n_{end}]}, \mathbf{p}_{goal}\|_2 & \text{If goal} \end{cases} \tag{11}$$

$$l_f[n_{end}] = ll[n_{end}] + lm[n_{end}] + lg[n_{end}] \tag{12}$$

As seen in (12), the final effective length of the tentacle is the summation of its actual length before hitting any obstacles (if any) and some modifiers based on the desired behavioral actions. Therefore the longest, and thus 'best' tentacle will naturally become the one which best satisfies the behavioral criteria. The 'priority' of the behavioral actions can be adjusted by modifying the DiffWeight, TravWeight UnknWeight and GoalWeight parameters.



## REFERENCES


[1] F. von Hundelshausen, M. Himmelsbach, F. Hecker, A. Mueller, and H.-J. Wuensche, "Driving with tentacles: integral structures for sensing and motion," *Journal of Field Robotics*, vol. 25, no. 9, pp. 640–673, 2008, ISSN: 1556-4967. DOI: 10.1002/rob.20256. [Online]. Available: http://dx.doi.org/10.1002/rob.20256.

[2] R. Brooks, "A robust layered control system for a mobile robot," *Robotics and Automation, IEEE Journal of*, vol. 2, no. 1, pp. 14 –23, 1986, ISSN: 0882-4967. DOI: 10.1109/JRA.1986.1087032.

[3] M. Manz, M. Himmelsbach, T. Luettel, and H.-J. Wuensche, "Fusing lidar and vision for autonomous dirt road following," English, in *Autonome Mobile Systeme 2009*, ser. Informatik aktuell, R. Dillmann, J. Beyerer, C. Stiller, J. Zllner, and T. Gindele, Eds., Springer Berlin Heidelberg, 2009, pp. 17–24, ISBN: 978-3-642-10283-7. DOI: 10.1007/978-3-642-10284-4_3. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-10284-4_3.

[4] S. Quinlan and O. Khatib, "Elastic bands: connecting path planning and control," in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, 1993, 802 –807 vol.2. DOI: 10.1109/ROBOT.1993.291936.

[5] M. Mataric, "Integration of representation into goal-driven behavior-based robots," *Robotics and Automation, IEEE Transactions on*, vol. 8, no. 3, pp. 304 –312, 1992, ISSN: 1042-296X. DOI: 10.1109/70.143349.