Implementation of biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance

Alan Gomez, Samuel Parra, and Brennan Penfold

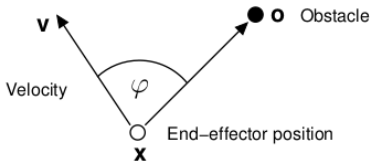

Fig. 1. Graphical representation of steering angle [1]

***Abstract*—**

I. Introduction

II. Approach

A. Distance to objects

B. Potential field

To avoid collisions we decided to implement the method presented in [1]. This method is based on dynamic movement primitives (DMP), which are used to generate a trajectory $x(t)$ with a velocity $v(t)$. These DMP are motivated from the dynamic of damped spring and can generate trajectories in many dimensions. These can be used to describe accelerations in 3D space that move towards a goal:

$$\dot{v} = K(g - x) - Dv - K(g - x_0)s + Kf(s) + p(x, v) \quad (1)$$

Where g is the goal, x is the current position, x_0 is the starting position, v is the current velocity, s is the phase variable, D is the damping constant, and K is the spring constant. The function $f(s)$ is a nonlinear function used to define a learned path for the manipulator to take. However, the terms dependent on s are not relevant for our application; thus we decided to not include it in the final equation.

The last term is the potential field generated by the obstacles. This term has a repel effect on the end effector, making it avoid obstacles. The potential field is calculated as follow: We first determine the steering angle φ using equation 2, which is the angle between the velocity vector and a vector from the end effector to the obstacle as seen in Fig 1. In our implementation, this vector is the shortest distance between the end-effector and the obstacle, as the original method considered points instead of volumes.

$$\varphi = cos^{-1}((o - v)^T v/(\|o - x\| \cdot \|v\|)); \quad (2)$$

This steering angle determines how sharp the end effector steers away from the obstacle. The closer the angle is to 0, the more abrupt is the change of direction. the potential field is given by:

$$p(x, v) = \gamma \sum_i R_i v \phi_i exp(-\beta \varphi_i) \quad (3)$$

Where γ and β are constants, and R is a rotation matrix which rotates by the axis $r = (o - x) \times v$ with an angle of rotation of $\pi/2$. This rotation matrix can be calculated with the Rodrigues' rotation formula [3]. The final potential field is the sum of the potential fields generated by all the obstacles. Our final equation converges towards the goal avoiding the obstacles in its way.

$$\dot{v} = K(g - x) - Dv + p(x, v) \quad (4)$$

C. Collision monitoring library

Our library was designed to be platform and framework independent. To achieve this, the library user must implement the Arm interface. This provides the flexibility of choosing how to describe mathematically the manipulator and update its position. Consider the case of two developers, one wants to use KDL for the forward and backwards kinematics, while the other one wants to derive the equations manually for a closed form solution. Both developers are supported by this library.

In order to monitor the distances from the links of the arm to the obstacles in the workspace we represent the geometry of the links with the use of capsules. This representation enables us to calculate the distances between the links themselves in order to monitor self-collisions along with regular obstacles. This requires that the developer updates the geometric representation alongside the arms movement in the Arm implementation.

D. Controller

The manipulator is controlled with a control loop that is executed at a specified rate. This loop is responsible for updating the geometric representation to match the real life manipulator and calculate the new velocity with the collision avoidance algorithm. In the loop the developer must keep the current position, velocity, positions of obstacles, and goal updated for the algorithm to produce an accurate output.

III. Experiments

A. Testing

B. ROS

C. Results

IV. Use case

A. Single arm

B. Dual arm

C. Self collision

V. Conclusion

Appendices

Acknowledgments

References

[1] G. O. Young, Synthetic structure of industrial plastics (Book style with paper title and editor), in Plastics, 2nd ed. vol. 3, J. Peters, Ed. New York: McGraw-Hill, 1964, pp. 1564.

[2] W.-K. Chen, Linear Networks and Systems (Book style). Belmont, CA: Wadsworth, 1993, pp. 123135.

[3] Belongie, Serge. "Rodrigues' Rotation Formula." From MathWorld–A Wolfram Web Resource, created by Eric W. Weisstein. https://mathworld.wolfram.com/RodriguesRotationFormula.html