Formation Control and Obstacle Avoidance in Second-Order Multi-agent Systems

Zihao Yan (40066713)
Electrical and Computer Engineering
Concordia University

Abstract— **In this report, a method of combining second-order formation control with path planning is introduced. In section 2, fundamentals of graph theory and artificial potential field are illustrated. Due to a second-order consensus, the vehicles exchange information according to a pre-specified communication digraph. In section 3, We use system dynamic of one agent and laplacian matrix to generate the consensus protocol. Moreover, a improved path planning method using artifitial potential field is developed. Numerical simulations are used to illustrate the results.**

I. INTRODUCTION

Group evacuation problem has long time been seeing as the basic implementation of multi-agent system. In a fore-seeable future, sending robots or group of robots to rescue people during crisis or devastating events would become more commonly used whatever the scenario is. Plenty of examples can be found in nature, birds flocking together to migrate, the ants form a line to seek for food,etc...All these examples which has been long existing on our planet shows that we can mimic how animals behave on the control of multi-agent system.

A fundamental problem of multi-agent control is the consensus, that is, all the agents or robots in a sensor network approach to reach a common value based on some algorithm. However, only reaching in a point does not seem to be practical enough in real-cased world. A modified consensus of an already existing consensus is the formation control.

Formation control requires that all the agents in the network try to maintain a certain pre-defined distance. which can be widely used in multiple fields. Regarding this, the position-based approach is more practical since it does not need agents to have high sensing capability and it is also easy to be simulated on software.

During the formation of all the agents, they need to avoid obstacles. One approach to do this is to generate APF(Artifitial Potential Field). By doing this, each agents will be dragged away from obstacles to the target directly.

II. PRELIMINARIES

A. Graph theory

In this section, basic concepts and results about graph theory is introduced. Let g = (G, E, V) be a weighted directed graph of order N,with the set of nodes $V = \{v_1, v_2, ..., v_N\}$,the set of directed edges E, and a weighted adjacency matrix $g = (G_{ij})_{N\times N}$. A directed edge e_{ij} is in this network is denoted by the ordered pair of nodes (v_i, v_j).

A directed path from node v_i to node v_j in g is a sequence of edges(v_i, v_j) in the directed network. A directed graph contains a directed spanning tree if and only if its subgraphs is a directed spanning tree.

A degree matrix denoted by $D \in R^N$ is a diagonal matrix that represents the number of links connected to each node or vertex. It is given by:

$$D_{i,j} = \begin{cases} deg(v_i), if \quad i = j \\ 0 \qquad otherwise \end{cases} \tag{1}$$

Similarly, an Adjacency matrix denoted by $A \in R^N$ is a square matrix that represents the information flow or connectivity between nodes(or between an agent and its neighbor). It is given by:

$$A_{i,j} = \begin{cases} 1 & ifjisneighborofi \\ 0 & otherwise \end{cases} \tag{2}$$

Then the laplacian matrix $L = D - A$.

Consider the dynamics of n agents is given by

$$\begin{aligned} \dot{x}(t) &= v(t) \\ \dot{v}(t) &= u(t) \end{aligned} \tag{3}$$

with $x(0) = x_0, v(0) = v_0$, where $x = [x_1, ...x_n]^T$ and $v = [v_1, ..., v_n]^T$.$x_i \in R$,v$_i \in R$, and $u_i \in R$ are the position, the velocity, and the force input, repectively, of the ith agent. or, equicalently:

$$u(t) = -Lx(t) - Lv(t) \tag{4}$$

for consensus protocol, following conditions has to be made, that is all the agent has to converge to the same states of its neighbors:

Definition 1. *Second-order consensus in multi-agent system can be achieved if for any initial conditions,*

$$\begin{aligned} &lim_{t\to\infty}||x_i(t)-x_j(t)||=0,\\ &lim_{t\to\infty}||v_i(t)-v_j(t)||=0,\\ &for \quad alli,j=1,2,...,N. \end{aligned} \tag{5}$$

because of (4), second-order consensus can be written as follows:

$$\begin{aligned} \dot{x}_i(t)&=v_i,\\ \dot{v}_i(t)&=-\alpha\sum_{j=1}^{N}L_{ij}x_j(t)-\beta\sum_{j=1}^{N}L_{ij}v_j(t),\\ i&=1,2,...,N \end{aligned} \tag{6}$$

where $x=(x_1^T,x_2^T,...,x_N^T)^T$, $v=(v_1^T,v_2^T,...,v_N^T)^T$, and $y=(x^T,v^T)^T$. Then the network can be rewritten in a compact matrix form as

$$\dot{y}(t)=(L\otimes I_n)y, \tag{7}$$

where L is laplacian matrix and $\otimes$ is the Kronecker product in Matlab.

B. Artificial Potential Field

In this section, some basic knowledges of adaptive artificial potential feld method is introduced. To implement potential field method, basic idea is to make the robot move towards the fastest decline direction of the total potential field by a gradient robot in the environment as a kind of robot moving in a virtual manual force field. The total force of the attractive force and the repulsive force controls the moving of the robot.

Assuming the position of robot is $P_r=(x,y)$ and the position of the goal is $P_g=(x,y)$, then attractive potential function is defined as:

$$U_a=w_a*(P_g-P_r)^2 \tag{8}$$

Anad the repulsive potential function is defined as:

$$U_{re}=\frac{1}{2}*\eta*(\frac{1}{\rho}-\frac{1}{\rho_0}) \tag{9}$$

Where: ρ is a positive constant, $\rho$is the shortest distance between robot and the obstacle, $\rho_0$is the largest impact distance of single obstacle.

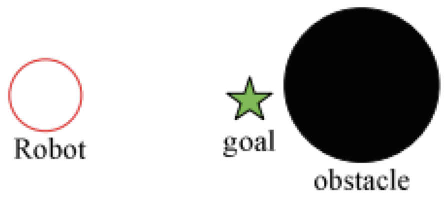


Fig.1 situation when goal is very close to obstacle.

However this algorithm has its limitation, that is, as is shown in Fig(1) when the position of goal is within the range of obstacles, the repulsive field rapidly increases and the attractive field decreases when the robot moves to the goal. Therefore, the robot can never reach the goal.

III. Formation Control For A Second-Order System And Improved Path Planning

A. Establish Continuous-time Model For Formation Control

Assume we give each agents with the same dynamics:

$$\dot{x}_i=Ax_i+Bu_i, i=1,...,N \tag{10}$$

where the x_i represents all the states that be stored in the vector, and u_i represents the control inputs. In the simulation below, for simplicity, we assume that $A=\begin{pmatrix}0&1\\0&1\end{pmatrix}$, which means in this system dynamic, the position is only determined by velocity, and velocity itself is only determined by the inputs.

In order to control the velocity and acceleration separately, we will use the notation $x_p=((x_p)_1,...,(x_p)_N)^T$,$x_v=((x_v)_1,...,(x_v)_N)^T$ to denote the vectors of position like and velocity like variables, so $x=x_p\otimes\begin{pmatrix}1\\0\end{pmatrix}+x_v\otimes\begin{pmatrix}0\\1\end{pmatrix}$.

Definition

Definition 2. *A formation is a vector $h=h_p\otimes\begin{pmatrix}1\\0\end{pmatrix}\in\mathbb{R}^{2nN}$*

(where $\otimes$ is the Kronecker product). The N vehicles are in formation h at time t if there are vectors $q,w\in R^n$ such that $(x_p)_i(t)-(h_p)_i=q$ and $(x_v)_i(t)=w$, for $i=1,...,N$. Inside the dynamics, we need all the agents to establish communication between each other. In order to do so, we implement laplacian which has been discussed before to build communication link between them. We define $\mathbb{J}_i$ as the set of neighbors of the ith agent. For further simplicity, the control u_i enables the feedback control law that each agent can use the relative information about its neighbors. We combine information is to define output function s y_i computed from an average of the relative displacements and velocities of the neighboring vehicles as follows:

$$y_i=(x_i-h_i)-\frac{1}{\mathbb{J}_i}\sum_{j\in\mathbb{J}_i}(x_j-h_j), i=1,...,N. \tag{11}$$

Then a slight modification has to be made that there is possibility that a vehicle might receive no information as well as the corresponding vertex in the graph would have no neighbors. Because of this, it is necessary to define the output function z_i by:

$$z_i=\sum_{j\in\mathbb{J}_i}((x_i-h_i)-(x_j-h_j)), i=1,...,N. \tag{12}$$

And the corresponding output vector can be written as $z=L(x-h)$, where L is the kronecher product of laplacian matrix.

When we collect all the equations into a single system we can obtain:

$$\begin{aligned} \dot{x}&=Ac*x+Bc*u,\\ z&=L(x-h) \end{aligned} \tag{13}$$

with $Ac=I_N\otimes A$, $B=I_N\otimes B$. The overall system dynamics in continuous time domain is:

$$\dot{x}=Ac*x+Bc*L*(x-h) \tag{14}$$

B. Digitization

In this part, the discretization of continuous time state space is discussed. The main purpose of digitization is that in continuous model, the $\dot{x}$ which is the differential state of the current state of all agents, we can only use ODE45 function in Matlab to solve this differential equation, which is always continuous and difficult to track and apply the outputs to the real cases. Suppose we are given the continuous time state space system:

$$\begin{aligned}\dot{x}(t) &= Ax(t)+Bu(t)\\ y(t) &= Cx(t)+Du(t)\end{aligned} \tag{15}$$

and apply an input that changes only at discrete sampling intervals. It would be nice if we could find matrices G and H, independent of t or k so that we could obtain a discrete time model of the system.

we start by using the solution of (13) to calculate t values of the state x at times KT and $(k+1)T$. These are:

$$x((k+1)T) = e^{A(k+1)T}x(0)+e^{A(k+1)T}\int_0^{(K+1)T} e^{-A\tau}Bu(\tau)d\tau \tag{16}$$

Let Ad be the A matrix of discrete-time model, $Ad = e^{At}$. from Tylor Expansion we know that:

$$e^{At} = I+tA+\frac{1}{2!}t^2A^2+\frac{1}{3!}t^3A^3+\ldots = \sum_{t=0}^{\infty}\frac{1}{i!}t^iA^i \tag{17}$$

if we abandon high order terms, we can get $Ad = I+tA$, $Bd = \int_0^t e^{Fy}Bdy$. The discretization of continuous-time model will be:

$$x((k+1)T) = Adx(kT)+Bdu(kT), \tag{18}$$

C. Improved Path Planning Algorithm

We consider the robot discrete-0time kinematic model is :

$$\begin{aligned}P_r(1,i+1) &= P_r(1,i)+\lambda * cos\gamma\\ P_r(2,i+1) &= P_r(2,i)+\lambda * sin\gamma\end{aligned} \tag{19}$$

where $P_r(1,i+1)$,$P_r(1,i)$ represent the robots' x-coordinate at time $i+1$. $P_r(2,i)$ represent the robots' y-coordinate values at time $i+1$ and time t

The goal potential plays the role of attractive potential, which is the same attractive potential function as the traditional artificial potential field method:

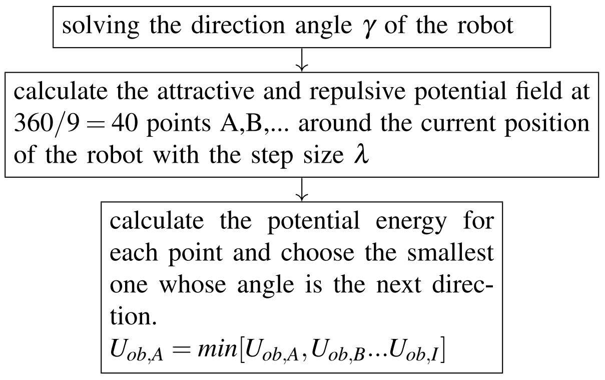
$$U_a = w_a * (P_g - P_r)^2 \tag{20}$$

With repulsive potential field, istead of using the traditional one, we use a Gaussian-like function as the obstacle potential:

$$U_{ob,i} = w_2 * exp(-\frac{1}{\sigma^2} * [(x_r - x_{ob,i})^2 + (y_r - y_{ob,i})^2 - R^2 - r^2]) \tag{21}$$

where x_r, y_r represent the robot coordinate values at the current time, $x_{ob,i}m, y_{ob,i}$ are the coordinate values of the ith obstacle, σ denotes the standard deviation of the obstacle potential, w_2 represent the obstacle potential scale factor,R is the size of the robot, r is the size of the obstacles. Since we do not consider the size of both obstacle and agents, the R and r here are meant to be 0.

The general method of path planning using artificial potential field is:


solving the direction angle γ of the robot
calculate the attractive and repulsive potential field at $360/9 = 40$ points A,B,... around the current position of the robot with the step size λ
calculate the potential energy for each point and choose the smallest one whose angle is the next direction.
$U_{ob,A} = min[U_{ob,A}, U_{ob,B} ... U_{ob,I}]$

IV. SIMULATIONS AND RESULTS

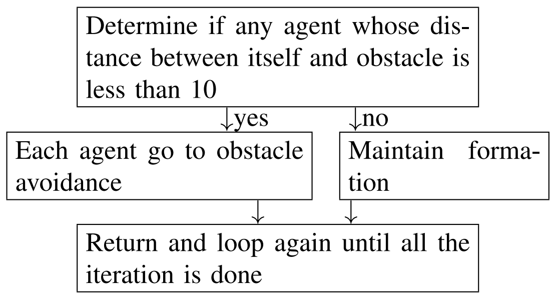
In this section, simulations are made through Matlab. Suppose we have the system dynamic for 4 agent:

$$\dot{x} = \begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}x+\begin{bmatrix}0\\ 1\end{bmatrix}u$$

The laplacian matrix for 4 agents is :

$$L = \begin{bmatrix}3 & -1 & -1 & -1\\ -1 & 3 & -1 & -1\\ -1 & -1 & 3 & -1\\ 0 & 0 & 0 & 0\end{bmatrix}$$

Where the leader does not follow the consensus, and its states are determined by pre-defined inputs.
Since all the states of agents are determined by the input. We use KRON function in matlab to produce canonical product for A in times of matrix I and kronecker product of B in times of L. By doing this we can build the continuous-time model that has been mentioned in (13). Then by digitization, the discrete-time model has been generated. After finalizing the control part of second-order consensus, we have to develop a general method for the overall control protocal for all the agents:


Determine if any agent whose distance between itself and obstacle is less than 10
yes
no
Each agent go to obstacle avoidance
Maintain formation
Return and loop again until all the iteration is done

By applying those control method, the simulation results are as follows:

The coordinates of obstacle and target are (50, 50) and (100, 100) respectively.

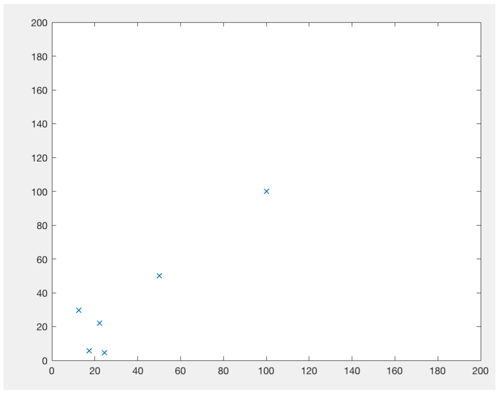


Fig.2 Start formation.

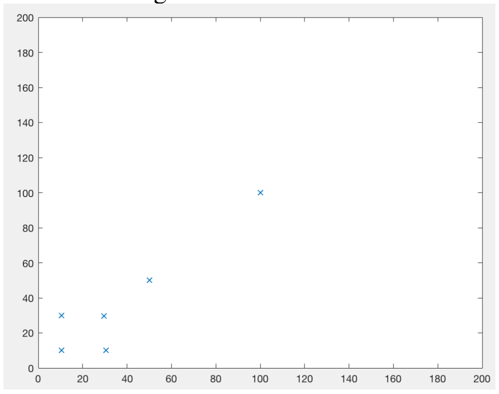


Fig.3 Formation completed

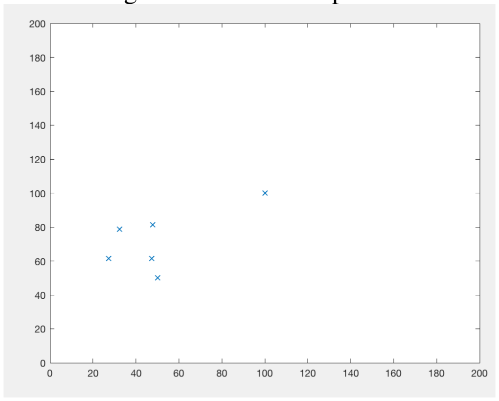


Fig.4 avoid obstacle.

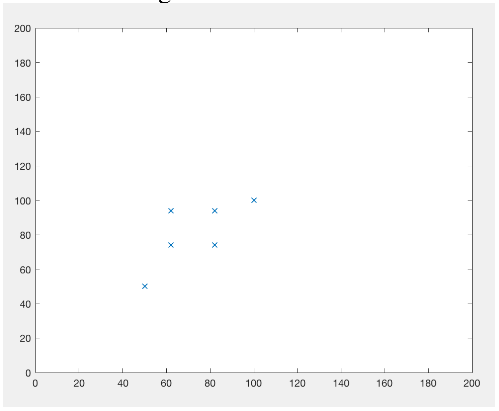


Fig.5 Back to formation

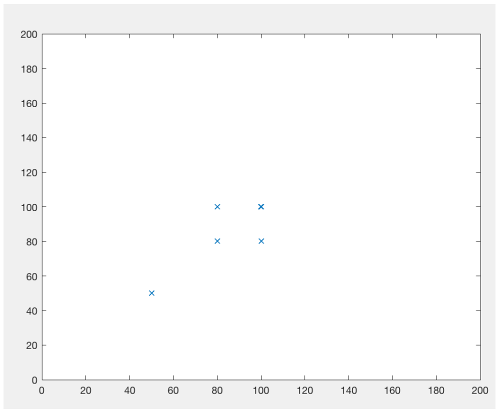


Fig.6 Achieve the goal

In Fig.1 the agents start to form a shape since there is no obstacle in the radius of 10. And in Fig.2 the formation is completed. Fig.3 shows that agents observe that there is an obstacle in radius, then they start to avoid it. After obstacle avoidance, in Fig.5 they go back to formation again. In Fig.6, they arrive their destination.

V. CONCLUSIONS

In this project, we tried to combine two popular algorithm together to realize the implementation. A second order formation consensus has been achieved by adding a pre-defined formation shape. The keys are to generate the control gain by using kronecker product in Matlab as well as the discretization. Also, an improved artificial potential field method is considered. Some parameters in potential field can be modified in the future to better the performance of potential field so that agent can be more responsive during obstacle avoidance. In the simulation part, the 4 agents continuous to follow a rectangular shape until one of them is getting to close to the obstacle. Then they start to seperate to obstacle avoidance on their own. Once they are out of the radius of 10 they recover to formation again until eventually hit the target. However, there are improvements can be done in the future, for instance, using second-order dynamics to avoid obstacles.

REFERENCES

[1] Mukherjee, Srijita, Formation Control of Multi-Agent Systems, hesis, August 2017; Denton, Texas.

[2] G. Lafferriere, A. Williams, J. Canghman, J.J.P. Veerman, Decentralized Control of Vehicle Formations.Portland State University, Portland, 2005.

[3] Li Zhou, Wei Li, Adaptive Artificial Potential Field Approach For Obstacle Avoidance Path Planning. Research Institute of Electronic Science and Technology of UESTC, Chengdu, China, 2014.

[4] Huan Pan, Xiaohong Nian, Consensus Analysis and Formation control of Second-order Multiagent Systems Via nonlinear Protocol, School of Physics Electrical Information Engineering, Ningxia University,China,2013.