Dissection of Modeling and Control of Formations of Nonholonomic Mobile Robots

Abhinav Bhamidipati
MEngg Robotics
University of Maryland
College Park, MD
Email: abhinav7@umd.edu

Sumedh Chinchmalatpure
MEngg Robotics
University of Maryland
College Park, MD
Email: sumedhc@umd.edu

***Abstract*—This paper addresses the control of a team of nonholonomic mobile robots navigating with obstacles while maintaining a desired formation and changing formations when required, using nonlinear control theory and graph theory. We use the terms $l{-}\psi$ and $l-l$, which are based on tracking the position and orientation of the robot relative to the leader, or the position of two leaders. Defining the concept of a transition matrix, we present an exhaustive list of all possible that can occur in the robot formation and the corresponding transition matrix column.**

***Index Terms*—Nonholonomic motion planning, Formation of mobile robots, graph theory, nonlinear control theory, transition matrix**

I. Introduction

In this paper, we discuss the tasks in which the robots are required to follow a trajectory while maintaining a desired formation and avoiding obstacles. We develop a framework for transitioning from one formation to another to avoid the obstacle. The team of robots in formation is modeled as a triple (g,r, H), where g is the gross position (x,y, θ) $\in SE(N)$ (N is dimensions, two or three most probably), r is a set of shape variables which describes the relative positions of the other robots in the team and H is the control graph which describes the control strategy used by each robot. In a situation like Fig.1 for example, the group of robots have to leave their formation to pass through the obstacles and regroup again to form the original formation again. [4][2]

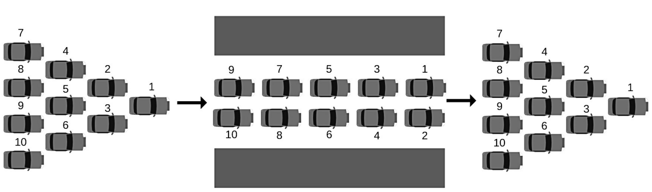

Fig. 1. Robots changing Formation

Previous works in motion planning focus on obtaining the path and, in some cases, designing feedback architecture and model-independent coordination strategy. If we consider a system of two mobile manipulators for cooperative manipulation and material handling, we would be able to incorporate geometric, kinematic, and dynamic constraints easily but we have a major setback. The main disadvantage of this method is that the number of variables in formulation increases the complexity of the planning task.[3] To overcome this complexity we describe two scenarios for feedback control in a formation. Let's assume some cases, In the first case, one robot follows another robot by controlling the relative distance and orientation between the two. This situation would be applicable to all the robots except for the lead robot as every other robot would have at least 1 leader robot. The second case would be the robot maintaining its position by maintaining a specified distance from the obstacle. To overcome these cases we will discuss about the control laws that can used in section II, we will then provide some enumerations and transitions in formations in section III, and provide with two example simulations in section IV.

II. Control Laws for Shape Variables

We now develop two types of feedback controllers for maintaining formation of a team of mobile robots.

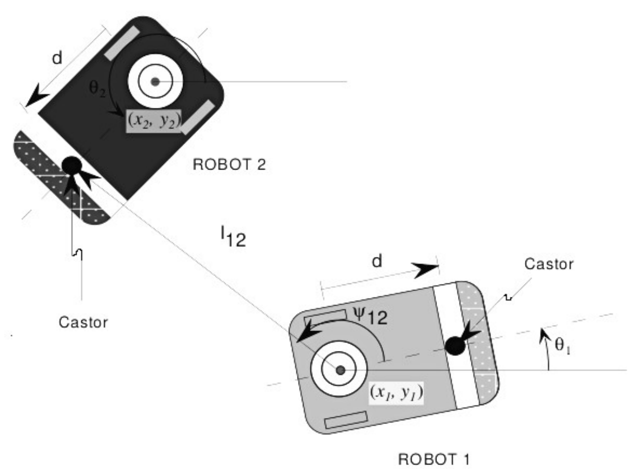

Fig. 2. Notation for $l-\psi$ control

Law 1 - $l{-}\psi$ control : Fig. 2, shows a system of two mobile robots separated by a distance of l_{12} between the center of

the first robot and the front castor of the second robot. The distance between the castor and the center of the axis of the wheels of each robot is denoted by d. Each robot has two actuated degrees of freedom. The state of the follower robot can be written relative to the lead robot as $(l_{12},\ \psi_{12},\ \theta_2)^T$.
The kinematic equation for the first robot in the system of two mobile robots is given by:

$$\dot{x}_i = v_i cos\theta_i$$
$$\dot{y}_i = v_i sin\theta_i$$
$$\dot{\theta}_i = \omega_i$$

The Kinematic equations for the second robot are:
The distance between the center of the first robot and the front castor of the second robot is:

$$\dot{l}_{12} = v_2 cos(\psi_{12} + \theta_1 - \theta_2) - v_1 cos\psi_{12} + d\omega_2 sin(\psi_{12} + \theta_1 - \theta_2)$$

as $\gamma_1 = \theta_1 + \psi_{12} - \theta_2$

$$\dot{l}_{12} = v_2 cos\gamma_1 - v_1 cos\psi_{12} + d\omega_2 sin\gamma_1$$

The relative orientation of the second robot with respect to l_{12}:

$$\dot{\psi}_1 2 = \frac{1}{l_{12}}[v_1 sin\psi_{12} - v_2 sin\gamma_1 + d\omega_2 cos\gamma_1 - l_{12}\omega_1]$$

Orientation of the second robot:

$$\dot{\theta}_2 = \omega_2$$

$v_i, \omega_i (i = 1, 2)$, are the linear and angular velocities at the center of the axle of each robot. In order to avoid collisions between robots, we will require that $l_{12} > d$.
We use standard techniques of input/output linearization to generate a control law that gives exponentially convergent solutions in the internal shape variables l_{12} and ψ_{12}. The control law is given by:[4]

$$\omega_2 = \frac{cos\gamma_1}{d}[\alpha_2 l_{12}(\psi_{12}^d - \psi_{12}) - v_1 sin\psi_{12} + l_{12}\omega_1 + \rho_{12} sin\gamma_1]$$
$$v_2 = \rho_{12} - d\omega_2 tan\gamma_1$$

where

$$\rho_{12} = \frac{\alpha_1(l_{12}^d - l_{12}) + v_1 cos\psi_{12}}{cos\gamma_1}$$

This leads to dynamics in the $l - \psi$ variables of the form:

$$\dot{l}_{12} = \alpha_1(l_{12}^d - l_{12})$$
$$\dot{\psi}_{12} = \alpha_2(\psi_{12}^d - \psi_{12})$$

Law 2 - $l - l$ control : Fig. 3. We have two non-holonomic mobile robots and the aim of this controller is to maintain the desired separation l_{13}^d and l_{23}^d between the follower and its two leaders. The state of the follower robot can be written relative to two leader robots a $(l_{13}, l_{23}, \theta_3)^T$. In this control, it requires regulating the desired lengths, l_{13}^d and l_{23}^d, of the third robot from its two leaders. The kinematic equations for the follower robot are expressed as

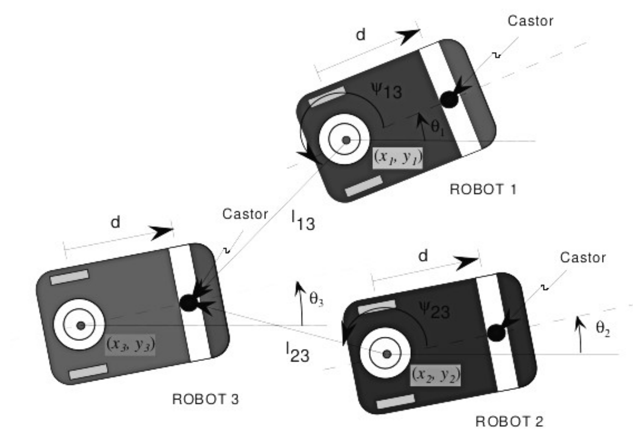

Fig. 3. Notation for $l - l$ control

The distance between the center of the first robot and the front castor of the third robot is:

$$\dot{l}_{13} = v_3 cos(\psi_{13} + \theta_1 - \theta_3) - v_1 cos\psi_{13} + d\omega_3 sin(\psi_{13} + \theta_1 - \theta_3)$$

as $\gamma_1 = \theta_1 + \psi_{13} - \theta_3$

$$\dot{l}_{13} = v_3 cos\gamma_1 - v_1 cos\psi_{13} + d\omega_3 sin\gamma_1$$

The distance between the center of the second robot and the front castor of the third robot is:

$$\dot{l}_{23} = v_3 cos(\psi_{23} + \theta_2 - \theta_3) - v_1 cos\psi_{23} + d\omega_3 sin(\psi_{23} + \theta_2 - \theta_3)$$

as $\gamma_2 = \theta_2 + \psi_{23} - \theta_3$

$$\dot{l}_{23} = v_3 cos\gamma_2 - v_1 cos\psi_{23} + d\omega_3 sin\gamma_2$$

Orientation of the third robot:

$$\dot{\theta}_3 = \omega_3$$

where, $\gamma_i = \theta_i + \psi_{i3} - \theta_3$($i$=1,2).
Again, we use input/output linearization to generate a feedback control law:

$$\omega_3 = \frac{1}{dsin(\gamma_1 - \gamma_2)}[\alpha_1(l_{13}^d - l_{13})cos\gamma_2 + v_1 cos\psi_{13} cos\gamma_2 - \alpha_2(l_{23}^d - l_{23})cos\gamma_1 - v_2 cos\psi_{23} cos\gamma_1]$$

$$v_3 = \frac{\alpha_1(l_{13}^d - l_{13}) + v_1 cos\psi_{13} - d\omega_3 sin\gamma_1}{cos\gamma_1}$$

This gives exponential convergence for the controlled variables:

$$\dot{l}_{13} = \alpha_1(l_{13}^d - l_{13})$$
$$\dot{l}_{23} = \alpha_2(l_{23}^d - l_{23})$$

These laws would be useful in maintaining 1) the desired separation and relative angle between the leader and the follower robot or 2) the desired separation of the follower robot from its two leaders.

III. Enumeration and Transitions in Formations

In this paper, we define *control graph* to be labeled digraphs with each vertex having a uniquely assigned integer number and following three constraints.
Constraint A: Every vertex of the robot has at least one incoming edge, except R_1, the lead robot with no incoming edges and at least one outgoing edge.
Constraint B: Every directed edge in the digraph goes from a lower vertex label to a higher vertex label.
Constraint C: The number of incoming edges for any vertex R_i ($i > 1$) is less than or equal to an integer $p \leq$ dim(SE(N)), that describes the number of output variables for a given robot.

A convenient method for representing graphs is through an n x n adjacency matrix. We can show that if there are n vertices in a control graph, there are exactly $M(n) = n!(n-1)!/2^{n-1}$ distinct control graphs (depending on the constraints stated above).

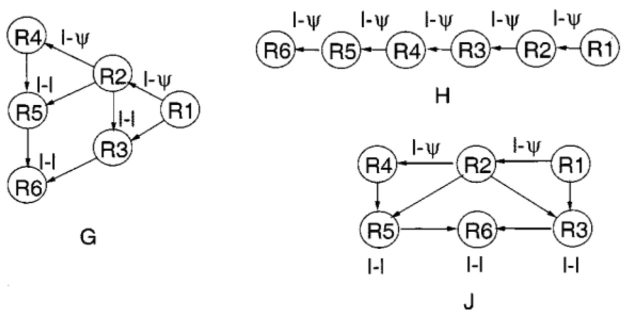

Fig. 4. Examples of formations.

We can classify control graphs based on the number of $l-\psi$ and $l-l$ controllers. As we have to count the number of possible graphs, we need to try the permutations of the robot's indices. We need some notation to derive these. Given n_ψ robots with $l-\psi$ controllers, let α represent the possible set of all robot indices (n_ψ) 1,...,n. We will get γ possible sets, α^1, α^2, . . , α^γ, where α^i and γ are

$$\alpha^i = \{N_1^i, N_2^i, ..., N_{n_\psi}^i\} = \{2, N_2^i, ..., N_{n_\psi}^i\}$$
$$\gamma = \frac{(n-2)!}{(N_\psi - 1)!(n - n_\psi - 1)!}$$

The index of the robot in $l-\psi$ control is denoted by $R_{N_j^i}$ $\forall$ $N_j^i \in$ 1,...,n. The set α uniquely determines a corresponding set β, the collection of the γ sets with remaining robot indices, denoted by β^1, β^2 ,..., β^γ where,

$$\beta^i = \{L_1^i, L_2^i, ..., L_{n_l}^i\}$$

The index of the robot formation in $l-l$ control is denoted with L_j^i.

Theorem 1: Given n robots in the formation with $n_\psi \geq 1$ robots having $l-\psi$ control and the remaining n_l = $(n - n_\psi - 1)$ robots having $l-l$ control, with $n_l > 0$, there are exactly T(n_ψ) control graphs where T(n_ψ) is given by

$$\mathrm{T}(n_\psi) = \frac{1}{2}^{n_l}(n-1)! \sum_{i=1}^{\gamma} \prod_{j=1}^{n_l} (L_j^i - 2)$$

Proof: R_1 being the lead robot in the formation, and R_2 always having a $l-\psi$ control, according to the constraint B, we get all the possible combinations of formation with n_ψ robots having $l-\psi$ and n_l robots having $l-l$ controllers in the sets of α and β. Considering a possible combination of α_i and β_i for $l-\psi$ and $l-l$ respectively. On taking L_j^i as the possible choices for the robot index for the leader robot, the total number of $l-l$ formations possible for these robots would be

$$\Omega^i = \prod_{j=1}^{n_l} \frac{(L_j^i - 1)(L_j^i - 2)}{2}$$

Similarly, for $l-\psi$ control of the robot, the choices for the leader from the set α_i will be $(N - j^i - 1)$. The total number of $l-\psi$ formations possible for these robots would be

$$\Psi^i = \prod_{j=1}^{n_\psi} (N_j^i - 1)$$

From the above two possible formations, we can find the total number of formations possible

$$\Theta^i = \Psi^i . \Omega^i$$

this computation being true to all sets of α^i and β^i

$$\begin{aligned}\Theta^i &= \Psi^i . \Omega^i \\ &= \prod_{j=1}^{n_\psi} (N_j^i - 1) . \prod_{j=1}^{n_l} \frac{(L_j^i - 1)(L_j^i - 2)}{2}\end{aligned}$$

Using the standard result for the product of consecutive integers

$$\prod_{j=1}^{n_\psi} (N_j^i - 1) = \frac{1}{2}^{n_\psi}(n_\psi - 1)!$$

Substituting this in the equation

$$\begin{aligned}\Theta^i &= \frac{1}{2}^{n_\psi}(n_\psi - 1)! \cdot \prod_{j=1}^{n_l} \frac{(L_j^i - 1)(L_j^i - 2)}{2} \\ &= \frac{1}{2}^{n_\psi}(n_\psi - 1)! \cdot \frac{1}{2^{n_l}} \cdot \prod_{j=1}^{n_l} (L_j^i - 1)(L_j^i - 2) \\ &= \frac{1}{2^{n_\psi + n_l}}(n_\psi - 1)! \cdot \prod_{j=1}^{n_l} (L_j^i - 1)(L_j^i - 2) \\ &= \frac{1}{2^{n_\psi + n_l}}(n_\psi - 1)! \cdot \sum_{i=1}^{\gamma} \prod_{j=1}^{n_l} (L_j^i - 2)\end{aligned}$$

as $n_\psi + n_l = n - 1$, we get the following equation

$$= \frac{1}{2}^n (n_\psi - 1)! \cdot \sum_{i=1}^{\gamma} \prod_{j=1}^{n_l} (L_j^i - 2)$$

Hence Proved.

Using this equation we will be able to easily compute all the possible sets of α^i and β^i. Taking an example n=3, we have two control graphs with two $l-\psi$ controllers and one control graph for one $l-\psi$ for R_2 and R_3 having a $l-l$ control. Therefore, we have three control graphs.

Theorem 1 allows us to classify control graphs but doesn't give any information about the equivalence classes of control graphs. An upper bound on the allowable unlabeled control graphs for n robots is given by the polynomial: [5]

$$Q_n(x) = \sum_{k=(n-1)}^{(2n-3)} a_k x^k$$

Based on the above equation, there are $(n-1)$ equivalence classes of digraphs, where the k^{th} equivalence class has a_k members, with a_k given by Polya's theorem. [1]
We can use this for order n = 3 to express the enumerating digraphs by the polynomial $Q_3(x)$:

$$Q_3(x) = 1 + x + 4x^2 + 4x^3 + 4x^4 + x^5 + x^6$$

The term $4x^2$ in the expression shows that there are 4 possible digraphs with 2 directed edges and these 4 digraphs belong to the same equivalence class. We can get the total number of digraphs of order 3 by summing the coefficients of the various powers of x in $Q_3(x)$, 1+1+4+4+4+1+1 = 16. So, the total digraphs for order n=3 is 16, $Q_3(1)$ = 16.

Polya's Theorem also provides us with a formula to count the number of distinct members of the set that remain unchanged under the group's transformation.

$$P_G(x_1, x_2, \ldots, x_k) = \frac{1}{|G|} \sum_{g \in G} x_1^{c_1(g)} x_2^{c_2(g)} \ldots x_k^{c_k(g)}$$

Here, $P_G(x_1, x_2, \ldots, x_k)$ is the cycle index polynomial of the group G, $x_1, x_2, \ldots, x_k$ are variables corresponding to the colors used, and $c_1(g), c_2(g), \ldots, c_k(g)$ are the cycle counts of the permutation g in G. The sum is taken over all elements of the group G, and $|G|$ is the order of the group.

The procedure for enumerating control graphs involves the following two steps

1) Using Polya's Theorem to enumerate all digraphs of order n $d_n(x) = 1 + a_1 x + a_2 x_2 + ... + a_{n(n-1)} x^{n(n-1)}$ The equation $Q_n(x) = \sum_{k=(n-1)}^{(2n-3)} a_k x^k$ provides a tighter bound on allowable digraphs.
2) Choose any control graph G, enumerated by $Q_n(x)$ and label the vertices arbitrarily. Using this labeling we can construct its adjacency matrix. Using the trial and error method, we can obtain the control graph which has an adjacency matrix that is upper triangular.

$Q-n(x)$varies from (n-1) to (2n-3), we have a range of graphs ranging from pure $l-\psi$ control to pure $l-l$ control, some combined $l-l$ and $l-\psi$ control. Fig. 5 shows an exhaustive list of all possible transitions in the control graph for the j^{th} robot, R_j $(j \neq 1)$.

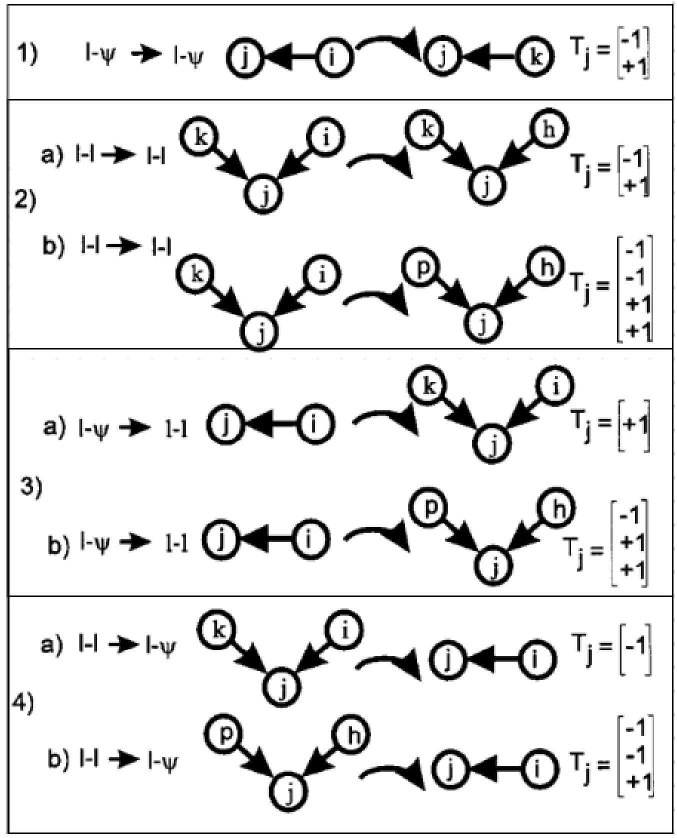

Fig. 5. Transition Laws

The transition from one control graph to another is modeled by a Transition matrix, T. Transition matrix is defined as the difference between the final and the initial adjacency matrices.

The first example from Fig.5 shows a transition from $l-\psi$ to $l-\psi$, the Matrix T_j shows -1 and +1 which shows that the vertex from i to j is broken and a new vertex from k to j is formed.
Similarly, the 2).b) example shows the transition from $l-l$ to $l-l$, where two vertices i and k are broken and new vertices p and h are formed to achieve the transition.
3).b) example shows transition from $l-\psi$ to $l-l$, where a vertex i and k is broken and new vertices p and h are formed to achieve the transition.
4).b) example shows transition from $l-l$ to $l-\psi$, where two vertices p and h are broken and new vertex i is formed to achieve the transition.

Consider an example from the Fig.6, where G is the initial control graph and H is the final control graph.

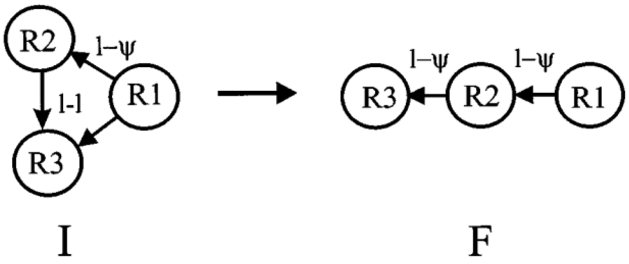

Fig. 6. Initial and Final Control Graph of Example 1

$$G = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \text{ and } H = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

The appearance of a 1 in a column for a robot defines its controller.

$$\sum_{column} 1's = \begin{cases} 0 \text{ Lead Robot} \\ 1 \text{ follower robot with } l-\psi \text{ control} \\ 2 \text{ follower robot with } l-l \text{ control} \end{cases}$$

Thus, T = H - G, is given by:

$$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

The appearance of -1 in the (1,3) entry denotes that the edge connecting the vertices 1 and 3 needs to be broken to achieve the transition. If +1 is present, it represents the formation of a new edge.

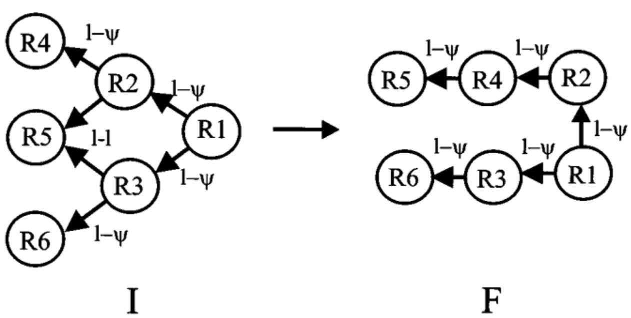

Fig. 7. Initial and Final Control Graph of Example 2

Consider an example from the Fig. 7, where I is the initial control graph and F is the final control graph.

$$I = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \text{ and } F = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The initial control graph has a single $l-l$ control follower robot, so only one column has 2 1s in the matrix. The final matrix has all follower robots with $l-\psi$ control, therefore every column has a single 1.

Thus, Transition Matrix T = F - I, is given by:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The appearance of -1 in the (1,5),(2,5) entry denotes that the edge connecting the vertices 1 and 5, 2 and 5 needs to be broken to achieve the transition. The appearance of +1 in the (3,5) entry denotes the formation of a new edge.

IV. SIMULATION RESULTS

We will try out two examples showing the transition from one formation to another based on the transitions enumerated in Fig. 5.

Example 1: We take the example of a triangular formation of robots, which then transitions into a straight-line formation. Fig. 6. shows the initial and final graph of this example. We derive the adjacency matrices from the initial and final positions of the robot and find the transition matrix. The transition matrix shows that there is a single change to be done to attain the final control graph. We then use the positions of the control graphs to plot the formation change in MATLAB. Fig. 8. shows the formation change of the robots from a triangular to a straight line in between the obstacles. The initial and final configurations of R_1 are from (0,0,0) to (0,0,6) as discussed in the introduction.[5]

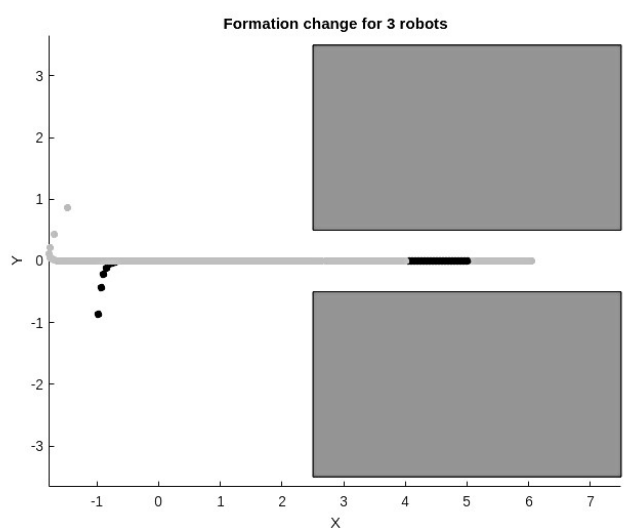

Fig. 8. Formation Change of 3 Robots

Example 2: The task here is to transition 6 robots from a triangular formation to a rectangular formation. Fig. 7. shows the initial and final control graphs of this example. We can see that there are changes as shown in the transition laws from Fig. 5. On finding the initial and final adjacency matrices, we find the transition matrix as done above. The transition matrix shows that there are three changes to be done to attain the final control graph, two vertices need to be broken and a vertex to be added. Fig. 9. shows the formation change of the 6 robots from a triangular formation to a rectangular formation avoiding the obstacle. The initial and final configurations of R_1 are from (0,0,0) to (0,0,5).

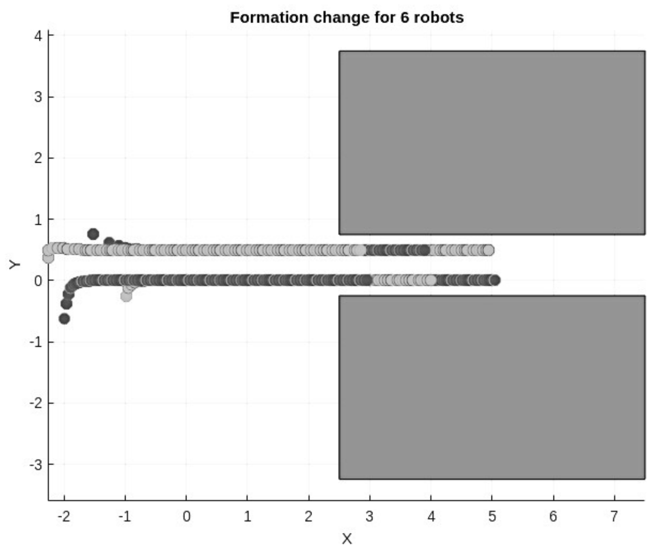

Fig. 9. Formation Change of 6 Robots

V. Conclusion

This paper has addressed the challenge of controlling the formation of mobile robots using a combination of non-linear control theory and graph theory. The main focus has been on breaking down the problem of controlling formations into the task of controlling a single robot's motion relative to one or two lead robots. This distinction is captured by the terms $l - \psi$ and $l - l$ control, representing control laws based on tracking the position and orientation of the robot relative to a single lead robot or the position relative to two lead robots, respectively.

The paper introduces the concept of transition matrices, derived from the initial and final control graphs, which provide valuable information about the addition or deletion of edges in the control graph. These matrices are used to present an exhaustive list of possible transitions within the robot formation, along with the corresponding transition matrix columns.

Two simulation examples are presented to demonstrate the application of the enumerated transitions. In the first example, a triangular formation of robots transitions into a straight-line formation, and in the second example, six robots transition from a triangular to a rectangular formation, both while avoiding obstacles.

Overall, the paper contributes to the understanding and development of strategies for controlling the formation of mobile robots in a flexible and adaptive manner. The combination of non-linear control techniques and graph theory provides a systematic framework for addressing the challenges associated with formation control, particularly in scenarios where robots need to adapt their formations to navigate obstacles or achieve specific tasks.

References

[1] Narsingh Deo. *Graph Theory with Applications to Engineering and Computer Science (Prentice Hall Series in Automatic Computation)*. Prentice-Hall, Inc., USA, 1974.

[2] J.P. Desai. Motion planning and control of cooperative robotic systems. In *Ph.D. dissertation, Univ. of Pennsylvania, Philadelphia*, Oct 1998.

[3] J.P. Desai and V. Kumar. Nonholonomic motion planning for multiple mobile manipulators. In *Proceedings of International Conference on Robotics and Automation*, volume 4, pages 3409–3414 vol.4, 1997.

[4] J.P. Desai, V. Kumar, and J.P. Ostrowski. Control of changes in formation for a team of mobile robots. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, volume 2, pages 1556–1561 vol.2, 1999.

[5] J.P. Desai, J. Ostrowski, and V. Kumar. Controlling formations of multiple mobile robots. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, volume 4, pages 2864–2869 vol.4, 1998.

Appendix

Listing 1. MATLAB Code for Example 1 Simulation

```matlab
function sim1()
    % Leader robot
    R1 = [0, 0];
    V1 = [0.1, 0];
    % Follower robots
    R2 = [-1, -sqrt(3)];
    R3 = [-1, sqrt(3)];

    desired_R2_pos = [-1, 0];
    desired_R3_pos = [-2, 0];

    dt = 0.5; % Time step
    t_final = 60; % End time

    obstacles = define_obstacles();

    % plot
    figure;
    hold on;
    axis equal;
    xlabel('X');
    ylabel('Y');
    title('Formation change for 3 robots');

    % plot obstacles
    plot_obstacles(obstacles);

    for t = 0:dt:t_final
        R1 = R1 + V1 * dt;

        R2 = update_follower_position(R1, R2, desired_R2_pos, dt);
        R3 = update_follower_position(R1, R3, desired_R3_pos, dt);

        plot(R1(1), R1(2), 'co', 'MarkerSize', 5, 'MarkerFaceColor', 'c');
        plot(R2(1), R2(2), 'ko', 'MarkerSize', 5, 'MarkerFaceColor', 'k');
        plot(R3(1), R3(2), 'yo', 'MarkerSize', 5, 'MarkerFaceColor', 'y');

        drawnow;
    end

    hold off;
end

function Rf = update_follower_position(Rl, Rf, desired_pos_rel_to_leader, dt)
    % proportional controller
    Kp = 1;  % gain
    desired_pos = Rl + desired_pos_rel_to_leader;
    error = desired_pos - Rf;
    Vf = Kp * error;
    Rf = Rf + Vf * dt; % position update
end

function obstacles = define_obstacles()
    obstacles = [
        5, 2, 5, 3; % Upper obstacle
        5, -2, 5, 3 % Lower obstacle
    ];
end

function plot_obstacles(obstacles)
    for i = 1:size(obstacles, 1)
        rectangle('Position', [obstacles(i, 1) - obstacles(i, 3)/2, obstacles(i, 2) - obstacles(i, 4)/2, obstacles(i, 3), obstacles(i, 4)], 'FaceColor', [0.5, 0.5, 0.5]);
    end
end
```

Listing 2. MATLAB Code for Example 2 Simulation

```matlab
function sim2()
    % initial triangular formation
    R1 = [0, 0];
    R2 = [-1, 0.5];
    R3 = [-1, -0.5];
    R4 = [-2, 1];
    R5 = [-2, 0];
    R6 = [-2, -1];

    V1 = [0.05, 0];
    V2 = [0.05, 0];
    offset = 0.1;

    offset_R3 = [-1, 0];
    offset_R4 = [-1, 0];
    offset_R5 = [-1, 0];
    offset_R6 = [-1, 0];

    % Obstacles
    obstacles = define_obstacles();

    dt = 1;          % Time step
    t_final = 100;   % End time

    % plot
    figure;
    hold on;
    axis equal;
    grid on;
    xlabel('X');
    ylabel('Y');
    title('Formation change for 6 robots');

    % Plot the obstacles
    plot_obstacles(obstacles);

    for t = 0:dt:t_final
        R1 = R1 + V1 * dt;
        R2 = R2 + V2 * dt;
        R2(1) = R1(1) - offset;

        R3 = update_follower_position(R1, R3, offset_R3, dt);
        R4 = update_follower_position(R2, R4, offset_R4, dt);
        R5 = update_follower_position(R4, R5, offset_R5, dt);
        R6 = update_follower_position(R3, R6, offset_R6, dt);

        plot_robot(R1, 'r');
        plot_robot(R2, 'm');
        plot_robot(R3, 'y');
```

```
        plot_robot(R4, 'g');
        plot_robot(R5, 'c');
        plot_robot(R6, 'b');

        drawnow;

        % Stop the simulation
        if R6(1) > obstacles(1, 1) + obstacles(1, 3) / 2
            break;
        end
    end

    hold off;
end

function Rf = update_follower_position(Rl, Rf, offset, dt)
    % Simple proportional controller
    Kp = 0.5;  % gain
    desired_position = Rl + offset;
    error = desired_position - Rf;
    Vf = Kp * error;
    Rf = Rf + Vf * dt;
end

function obstacles = define_obstacles()
    obstacles = [
        5, 2.25, 5, 3; % Upper obstacle
        5, -1.75, 5, 3 % Lower obstacle
    ];
end

function plot_obstacles(obstacles)
    for i = 1:size(obstacles, 1)
        rectangle('Position', [obstacles(i, 1) - obstacles(i, 3)/2, obstacles(i, 2) - obstacles(i, 4)/2, obstacles(i, 3), obstacles(i, 4)], 'FaceColor', [0.5, 0.5, 0.5]);
    end
end

function plot_robot(position, color)
    plot(position(1), position(2), 'o', 'MarkerSize', 8, 'MarkerFaceColor', color);
end
```