2010 IEEE/ASME International Conference on
Advanced Intelligent Mechatronics
Montréal, Canada, July 6-9, 2010

Centralized and Decentralized Multi-Robot Control Methods using the Cluster Space Control Framework

Ignacio Mas and Christopher Kitts

***Abstract*— The *cluster space* control technique promotes simplified specification and monitoring of the motion of mobile multi-robot systems. Previous work has established the conceptual foundation of this approach and has experimentally verified and validated its use. In this publication, we summarize the definition of the *cluster space* framework for planar robots and study some of the most common formation control methods found in the literature from the *cluster space* perspective. In doing this, we show that our proposed formation control framework can be implemented in various ways; as a centralized or distributed system, and with different levels of scalability depending on the particular cluster definition chosen. In particular, leader-follower, potential functions and virtual structures approaches are analyzed with the intent of addressing the generality and flexibility of the *cluster space* formation control approach. Experimental results illustrate the different implementations which are then compared and contrasted.**

***Index Terms*— cluster space, multi-robot systems, formation control, distributed control.**

I. Introduction

Robotic systems offer many advantages to accomplishing a wide variety of tasks given their strength, speed, precision, repeatability, and ability to withstand extreme environments. Whereas most robots perform these tasks in an isolated manner, interest is growing in the use of tightly interacting multi-robot systems to improve performance in current applications and to enable new capabilities. Potential advantages of multi-robot systems include redundancy, increased coverage and throughput, flexible reconfigurability and spatially diverse functionality. For mobile systems, one of the key technical considerations is the technique used to coordinate the motions of the individual vehicles. A wide variety of techniques have been and continue to be explored, drawing on work in control theory, robotics, and biology [1] and applicable for robotic applications throughout land, sea, air, and space. Notable work in this area includes the use of leader-follower techniques, in which follower robots control their position relative to a designated leader [2], [3]. A variant of this is leader-follower chains, in which follower robots control their position relative to one or more local leaders, which, in turn, are following other local leaders in a network that ultimately is led by a designated leader. For example, a follower robot might position itself with respect to the local leader by sensing and controlling its relative distance and bearing to that leader, or by maintaining its relative distance between two local leaders [4]. Several approaches employ artificial attraction/repulsion fields as a construct to establish formation-keeping forces for individual robots within a formation. For example, potential fields may be used to implement repulsive forces among neighboring robots and between robots and objects in the field in order to symmetrically surround an object to be transported [5]. Potential fields and behavioral motion primitives have also been used to compute reactive robot drive commands that balance the need to arrive at the final destination, to maintain relative locations within the formation, and to avoid obstacles [6], [7]. As another example, the virtual bodies and artificial potentials (VBAPs) approach uses potential fields to maintain the relative distances both between neighboring robots as well as between robots and reference points, or 'virtual leaders', that define the 'virtual body' of the formation [8], [9]. This approach has been successfully demonstrated in field tests of underwater robots performing a distributed sampling mission [10].

The motivation of the proposed *cluster space* approach is to promote the simple specification and monitoring of the motion of a mobile multi-robot system. This strategy conceptualizes the n-robot system as a single entity, a *cluster*, and desired motions are specified as a function of cluster attributes, such as position, orientation, and geometry. These attributes guide the selection of a set of independent system state variables suitable for specification, control, and monitoring. These state variables form the system's *cluster space*. The particular selection of such variables play an essential role in the dynamic behavior of the system as well as in the different resulting control architectures, for example, in centralized or distributed approaches.

Previous work presented a generalized framework for developing the *cluster space* approach for a system of n robots, each with m degrees of freedom (DOF)[11]. This framework has been successfully demonstrated for both holonomic and non-holonomic centralized systems with limited numbers of rovers, including automated trajectory control [12] and potential field-based obstacle avoidance [13]. The method has also been implemented for formations of marine surface

This work has been sponsored through a variety of funding sources to include Santa Clara University Technology Steering Committee grant TSC131 and National Science Foundation Grant No. CNS-0619940.

The authors are with the Robotic Systems Laboratory, Santa Clara University, 500 El Camino Real, Santa Clara, CA 95053, USA iamas,ckitts@scu.edu

vessels [14] and with air blimps [15].

In this paper, we first briefly review the cluster space paradigm and then we study some of the most common formation control techniques and how they can be defined as particular cases within the *cluster space* control framework. This interpretation is convenient for our development of the generalized framework and we acknowledge that different interpretations of the same techniques may also be valid.

II. Cluster Space Framework for Planar Robots

The first step in the development of the *cluster space* control architecture is the selection of an appropriate set of *cluster space* state variables. To do this, we introduce a cluster reference frame and select a set of state variables that capture key pose and geometry elements of the cluster.

Consider the simplified, general case of a system of n planar robots where each robot has 3-DOF (two translational and one rotational) and an attached body frame, as depicted in Fig. 1.

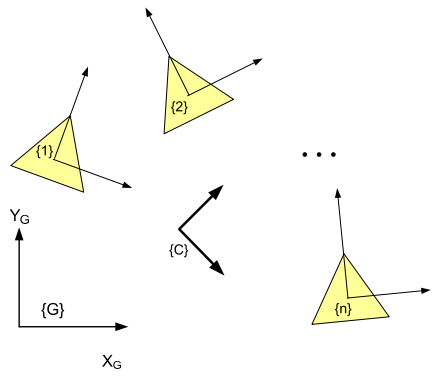

Fig. 1. Cluster and robot frame descriptions with respect to a global frame

Typical robot-oriented representations of pose use $3n$ variables to represent the position and orientation of each of the robot body frames, $\{1\},\{2\},\cdots,\{n\}$, with respect to a global frame $\{G\}$. In contrast, consideration of the *cluster space* representation starts with the definition of a cluster frame $\{C\}$, and its pose. The pose of each robot is then expressed relative to the cluster frame. We note that the positioning of the $\{C\}$ frame with respect to the n robots is often critical in achieving a *cluster space* framework that benefits the operator/pilot. In practice, $\{C\}$ is often positioned and oriented in a manner with geometric significance, such as at the cluster's centroid and oriented toward the 'lead' vehicle or alternatively, coincident with a lead vehicle's body frame.

A. Selection of Cluster Space Variables

We select as our state variables a set of position variables (and their derivatives) that capture the cluster's pose and geometry. For the case of 3-DOF robots, where the pose variables of $\{C\}$ with respect to $\{G\}$ are (x_c, y_c, θ_c) and where the pose variables for robot i with respect to $\{G\}$ are (x_i, y_i, θ_i) for $i = 1, 2, \cdots, n$, the *cluster* pose and geometry variables are

$$\begin{aligned} c_1 &= f_1(x_c, y_c, \theta_c, x_1, y_1, \theta_1, \cdots, x_n, y_n, \theta_n) \\ c_2 &= f_2(x_c, y_c, \theta_c, x_1, y_1, \theta_1, \cdots, x_n, y_n, \theta_n) \\ &\vdots \\ c_{3n} &= f_{3n}(x_c, y_c, \theta_c, x_1, y_1, \theta_1, \cdots, x_n, y_n, \theta_n). \end{aligned} \quad (1)$$

The appropriate selection of cluster state variables may be a function of the application, the systems design, and subjective criteria such as operator preference. The interdependency of the cluster parameters and the robot parameters are a key characteristic that defines the ability to operate the system in a centralized or distributed fashion. When the robot level parameters are highly coupled through the cluster level parameters, the systems tends to a centralized architecture and the formation can be tightly controlled. When the parameters are loosely coupled, the definition allows for decentralized implementations at the cost of inferior formation shape control.

B. Cluster Kinematic Relations

We wish to specify multi-robot system motion and compute required control actions in the *cluster space* using cluster state variables selected as described in the previous section. Given that these control actions will be implemented by each individual robot (and ultimately by the actuators within each robot), we develop formal kinematic relationships relating the *cluster space* variables and robot space variables.

We can therefore define $3n \times 1$ robot and cluster pose vectors, $\vec{r}$ and $\vec{c}$, respectively. These state vectors are related through a set of forward and inverse position kinematic relationships:

$$\begin{aligned} \vec{c} &= \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_{3n} \end{pmatrix} = KIN(\vec{r}) = \\ &= \begin{pmatrix} g_1(r_1, r_2, \cdots, r_{3n}) \\ g_2(r_1, r_2, \cdots, r_{3n}) \\ \vdots \\ g_{3n}(r_1, r_2, \cdots, r_{3n}) \end{pmatrix} \end{aligned} \quad (2)$$

$$\vec{r} = \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_{3n} \end{pmatrix} = INVKIN(\vec{c}) =$$

$$= \begin{pmatrix} h_1(c_1, c_2, \cdots, c_{3n}) \\ h_2(c_1, c_2, \cdots, c_{3n}) \\ \vdots \\ h_{3n}(c_1, c_2, \cdots, c_{3n}) \end{pmatrix}, \quad (3)$$

where $KIN()$ and $INVKIN()$ denote the forward and inverse position kinematic transforms, respectively.

We may also consider the formal relationship between the robot and *cluster space* velocities, $\dot{\vec{r}}$ and $\dot{\vec{c}}$. From (2), we may compute the differentials of the *cluster space* state variables, c_i, and develop a Jacobian matrix, $J(\vec{r})$, that maps robot velocities to cluster velocities in the form of a time-varying linear function:

$$\dot{\vec{c}} = J(\vec{r})\ \dot{\vec{r}}, \quad (4)$$

where J is the matrix of all first-order partial derivatives of the position kinematic functions.

In a similar manner, we may develop the inverse Jacobian, $J^{-1}(\vec{c})$, which maps cluster velocities to robot velocities. Computing the robot space state variable differentials from (3) yields:

$$\dot{\vec{r}} = J^{-1}(\vec{c})\ \dot{\vec{c}}, \quad (5)$$

where J^{-1} is the matrix of all first-order partial derivatives of the inverse position kinematic functions.

The Jacobian and inverse Jacobian matrices can be used as indicators of the level of coupling between the cluster parameters. Zero elements in the matrices reflect no coupling and non-zero elements show tight dependencies between parameters. Therefore, sparse matrices result in definitions convenient for distributed architectures and highly populated matrices result in centralized implementations.

C. Control Framework

Fig. 2 presents the general control architecture for trajectory based *cluster space* control for an n-robot system. The cluster level controller compares the cluster state with desired trajectory values and outputs cluster commanded velocities, which are translated into individual robot velocities through the inverse Jacobian. Data from the robots are converted to *cluster space* information through the forward kinematics and Jacobian and fed back into the controller to close the loop. This diagram intends to show the flow of signals and does not necessarily reflect the physical distribution of functionality. Each block may be implemented in one or more of the different physical components of the system.

This approach considers a basic kinematic model of the system [16] where $\dot{\vec{r}} = \dot{\vec{r}}_{cmd}$, or multiplying both sides of the equation by the Jacobian matrix as in (4),

$$\dot{\vec{c}} = \dot{\vec{c}}_{cmd}, \quad (6)$$

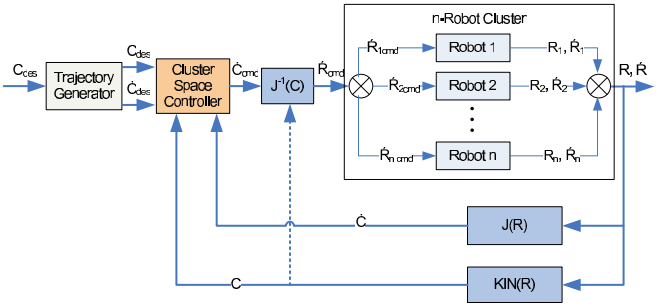

Fig. 2. Cluster space control architecture for a mobile n-robot system. In this cluster space control architecture, desired motions and control actions are computed in cluster space; control actions are converted to robot space through the use of the inverse Jacobian relationship.

based on the assumption that each robot is capable of closed-loop velocity control in order to cancel existing dynamics [17], a level of functionality typically built into a variety of commercially-available robotic platforms.

III. Case Study 1: Cluster Space Representation of a Three-Robot Centralized System

To illustrate the centralized approach, we have selected a particular planar cluster formed by three mobile robots.

A. Cluster Space State Variable Selection

Fig. 3 depicts the relevant reference frames for the planar 3-robot problem. We have chosen to locate the cluster frame {C} at the cluster's centroid, oriented with Y_c pointing toward Robot 1.

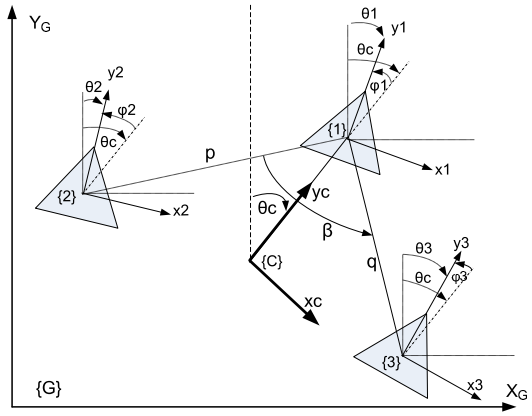

Fig. 3. Reference frame definition placing the cluster center at the triangle centroid

Given the parameters defined by Fig. 3, the robot space pose vector is defined as:

$$\vec{r} = (x_1, y_1, \theta_1, x_2, y_2, \theta_2, x_3, y_3, \theta_3)^T, \quad (7)$$

where $(x_i, y_i, \theta_i)^T$ defines the position and orientation of robot i. The *cluster space* pose vector definition is given by:

$$\vec{c} = (x_c, y_c, \theta_c, \phi_1, \phi_2, \phi_3, p, q, \beta)^T, \quad (8)$$

where $(x_c, y_c, \theta_c)^T$ is the cluster position and orientation, ϕ_i is the yaw orientation of rover i relative to the cluster, p and q are the distances from rover 1 to rover 2 and 3, respectively, and β is the skew angle with vertex on rover 1. Based on this, the nine robot space state variables (three robots with three DOF per robot) are mapped into nine *cluster space* variables for a nine DOF cluster.

Given the aforementioned selection of *cluster space* state variables, we can express the forward and inverse position kinematics of the three-robot system. These equations and their respective Jacobian matrices are presented in detail in [11].

It should be noted that most of the variables depend on all the robot level variables in order to be computed. The state of all robots are necessary to reconstruct the cluster variables which in turn affect all the robot variables when modified. This results in a highly populated Jacobian matrix, which indicates that this definition is essentially centralized.

IV. Case Study 2: Cluster Space Representation of a Lead-Follower Distributed System

The selection of the cluster variables can also result in a very different set of kinematic transforms that allows for variations in the implementation of the control architecture. In this section, we study the lead-follower approach, one of the most common formation control methods [2], [3], from the cluster space perspective.

To do this, we select the cluster variables as follows [18]: the cluster frame is located at the lead robot frame and for the follower robot i, where $i = 2, 3, ..., n$, the relative position with respect to the lead is defined by the relative distance and bearing, d_{1i} and ψ_{1i} respectively. The complete set of variables is obtained by adding n additional robot orientation variables ϕ_i which coincide with the robot space orientations θ_i. Fig 4 shows these variables for a three-robot cluster. The *cluster space* pose vector definition is then

$$\overrightarrow{c} = (x_c, y_c, \phi_c, d_{12}, \psi_{12}, \phi_2, ..., d_{1n}, \psi_{1n}, \phi_n)^T . \quad (9)$$

Given this cluster selection, the forward kinematics for the cluster position are trivial, $x_c = x_1$, and $y_c = y_1$. For the follower robot i, the transforms are:

$$d_{1i} = \sqrt{(x_1 - x_i)^2 + (y_1 - y_i)^2} \quad (10)$$

$$\psi_{1i} = atan2\,(y_1 - y_i, x_1 - x_i)\,. \quad (11)$$

The inverse position kinematics for the follower robot i are therefore defined by:

$$x_i = x_c + d_{1i}\cos(\psi_{1i}) \quad (12)$$

$$y_i = y_c - d_{1i}\sin(\psi_{1i})\,. \quad (13)$$

It can be seen from the equations that the transforms for the lead robot are independent of the state of other robots in the formation, and the transforms for the follower robots are only a function of their state and that of the lead. This allows for a decoupled system that can be divided and implemented in a decentralized fashion. This also offers a high level of scalability as adding new robots to the formation adds new cluster parameters that do not affect the existing ones. Furthermore, the forward Jacobian matrix will be of the form:

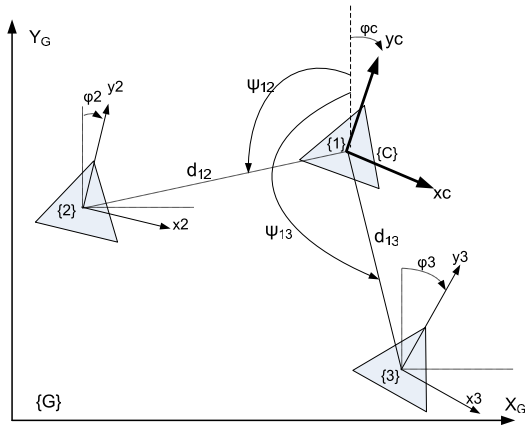

Fig. 4. Reference frame definition for lead-follower configuration

$$J(\overrightarrow{r}) = \begin{pmatrix} I & 0 & 0 & \cdots & 0 \\ A & B & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A & 0 & 0 & \cdots & B \end{pmatrix}, \quad (14)$$

where I is the identity matrix and the blocks A and B repeat along the matrix.

The inverse Jacobian matrix will be of the form:

$$J^{-1}(\overrightarrow{c}) = \begin{pmatrix} I & 0 & 0 & \cdots & 0 \\ I & C & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ I & 0 & 0 & \cdots & C \end{pmatrix}. \quad (15)$$

The Jacobian and inverse Jacobian matrices are highly sparse and can be divided into blocks to be computed locally at each robot in the formation depending on the role they assume as leader or follower. For the leader, the Jacobian and inverse Jacobian are the identity matrix I and the space transformation is trivial. For each follower i, the robot space velocities can be computed as follows:

$$\begin{pmatrix} \dot{x_i} \\ \dot{y_i} \\ \dot{\theta_i} \end{pmatrix} = \begin{bmatrix} A & B \end{bmatrix} \begin{pmatrix} \dot{x_c} \\ \dot{y_c} \\ \dot{\phi_c} \\ \dot{d_{1i}} \\ \dot{\psi_{1i}} \\ \dot{\phi_i} \end{pmatrix}, \quad (16)$$

and the cluster space velocities are:

$$\begin{pmatrix} \dot{d}_{1i} \\ \dot{\psi}_{1i} \\ \dot{\phi}_i \end{pmatrix} = \begin{bmatrix} I & C \end{bmatrix} \begin{pmatrix} \dot{x}_1 \\ \dot{y}_1 \\ \dot{\theta}_1 \\ \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{pmatrix}. \tag{17}$$

We can now lay out a distributed architecture for the cluster space lead-follower controller. To do this, the controller is distributed into subgroups each of which controls the cluster parameters corresponding to a specific robot. The controllers of the follower robots, together with the transforms (16) and (17), perform the closed loop cluster calculations. Each controller subgroup makes use of its host follower robot state and that of the lead (or its local lead). Therefore, no global communication is required to control the formation. Fig. 5 shows the distributed cluster control architecture for a 3-robot formation where robot 1 is the leader and robots 2 and 3 are followers.

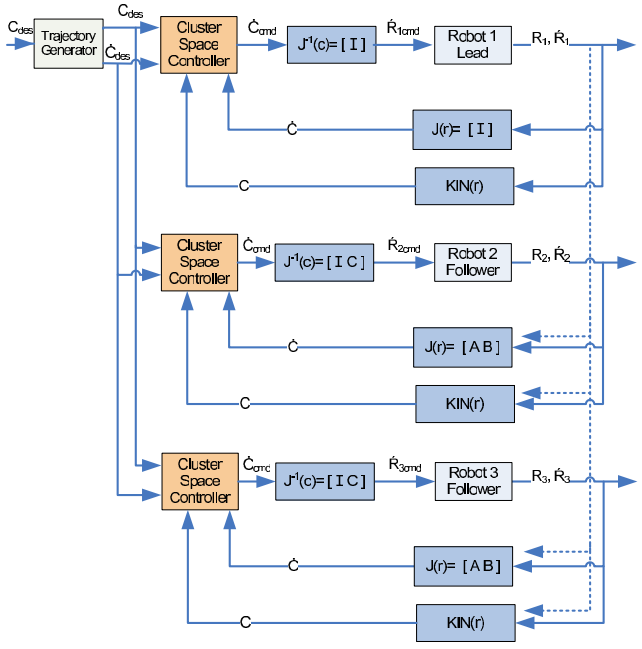

Fig. 5. Decentralized cluster space control architecture for a lead-follower control definition. In this architecture, followers only need knowledge of their state and that of the lead robot to compute the control effort.

V. Case Study 3: Cluster Space Representation of a Potential Fields Based Centralized System

Another well-known technique for formation control is the 'Potential Fields' method, where potential functions are defined in order to create virtual forces [7], [19]. These forces define attractions and repulsions that position robots in desired locations with respect to other robots or obstacles in the environment. Goal positions also attract robots to produce navigation trajectories.

This method can also be considered from a cluster space perspective. The key parameters to this approach are the relative position of the robots and the distances of the robots to the desired goal position (or to obstacles for repulsive forces). For an n-robot formation, the cluster parameters chosen are:

$$\overrightarrow{c} = (d_{12}, d_{13}, \ldots, d_{1n}, d_{1g}, d_{23}, d_{24}, \ldots, d_{2n}, d_{2g}, \ldots)^T, \tag{18}$$

where $d_{ji} = (x_{ji}, y_{ji})$ are the relative distances of robot j with respect to robot i and $d_{jg} = (x_{jg}, y_{jg})$ are the relative distances from robot j to the goal position G. The goal position can also be interpreted as a virtual leader as defined in [9]. These parameters are indicated in Fig. 6 for a 3-robot system. It should be noted that d_{ij} and d_{ji} may also be taken as different variables to facilitate a decentralized implementation where each robot only makes use of its own range sensors to estimate its neighbor's positions.

The number of cluster parameters may exceed the DOF of the system. If that is the case, the operator must make sure that the specification of the system is consistent, for example $d_{ji\ desired} = -d_{ij\ desired}$.

The kinematic transforms in this case are simple. The cluster parameters are the differences between robot positions. The Jacobian matrix will have the form:

$$\begin{pmatrix} \dot{d}_{ij} \\ \dot{d}_{ik} \\ \dot{d}_{jk} \end{pmatrix} = \begin{bmatrix} -I & I & 0 \\ -I & 0 & I \\ 0 & -I & I \end{bmatrix} \begin{pmatrix} \dot{d}_i \\ \dot{d}_j \\ \dot{d}_k \end{pmatrix}, \tag{19}$$

where $\dot{d}_i = (\dot{x}_i, \dot{y}_i)^T$ is the velocity of robot i in robot space. The Jacobian matrix can also be divided in blocks to implement a decentralized controller where each block computes motions as a function of relative distances to the relevant neighbors.

The cluster controller then uses potential functions to produce commanded velocities as a function of the cluster parameters. The potential functions are [7]:

$$U_{ij} = \frac{1}{2}\xi \left\| d_{ij} - d_{ij\ desired} \right\|^2. \tag{20}$$

The sum of the individual potential functions over the cluster parameters result in the total potential function $U(\overrightarrow{c})$. The force producing the motion is $F = -\nabla U(\overrightarrow{c})$, which defines the control law. The resulting control architecture is shown in Fig. 7.

Other implementations of potential field methods can be found in the literature. In [20], the Euclidean distance between robots (or norm of d_{ij}) is used as the cluster parameter that defines the forces between robots. This produces a formation with an unconstrained orientation. In [9], the same approach

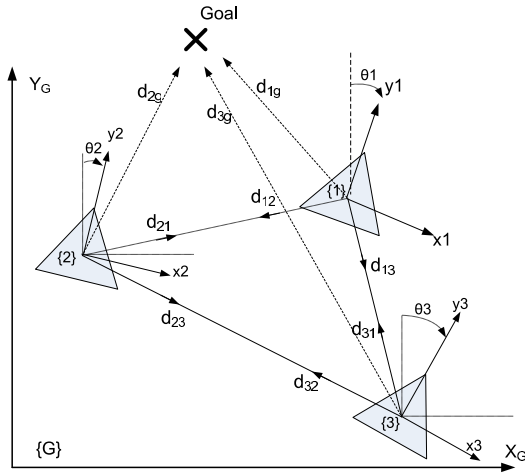

Fig. 6. Reference frame definition for potential field forces configuration

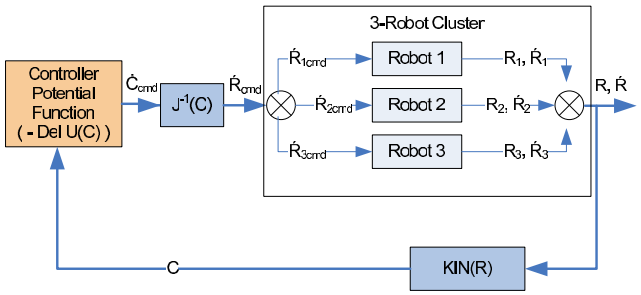

Fig. 7. Cluster space control architecture for a potential field driven control definition.

based on Euclidean distances is used and the problem of unconstrained formation rotation is solve with additional 'virtual leaders', which from our perspective can be considered additional goal positions.

VI. Case Study 4: Cluster Space Representation of a Virtual Structure System

This method uses the idea that points in space maintaining fixed geometric relationships are actually behaving in the same way as points on a rigid body moving through space. If robots behaved in this way, they would be moving inside of a virtual structure (VS) [8].

In [8], a general algorithm for virtual structure control is presented:

1) Align the VS with the current robots- VS alignment is via minimization of error between the virtual structure and the current robot position.
2) Move the VS by a Δx and $\Delta\theta$. This increment is determined by a local or global trajectory generator.
3) Compute individual robot trajectories to intercept the desired VS point.
4) Adjust wheel velocities to follow the desired trajectories
5) Goto 1.

This same algorithm can be rewritten under the cluster space control perspective with minimal semantic modifications as follows:

1) Compute the cluster forward kinematics to obtain the cluster state parameters which define the current virtual structure.
2) Define the desired cluster space trajectory (shape parameters and structure position).
3) Compute cluster space commands needed to minimize errors in cluster space (cluster controller).
4) Transform cluster space commands to robot space command using the cluster inverse Jacobian an apply commands to robots.
5) Goto 1 to close the loop.

The cluster space parameters define the virtual structure and commands are generated in cluster space to keep the virtual structure rigid. A formal set of transforms converts the cluster commanded velocities to robot specific velocities and allows to calculate the state of the virtual structure as a function of the robot positions in order to close the control loop.

From the architecture point of view, this case gets reduced to the potential functions method, where relative robot positions are controlled and a goal position trajectory moves the formation.

VII. Experimental Results

The different architectures were experimentally tested in order to demonstrate the functionality of the approaches and compare their behaviors. A testbed consisting of three Pioneer robots with custom sensor suites based on Ultra-Wide Band technology was utilized [12].

A. Centralized System Implementation

The system of Section III is implemented in a centralized fashion and the experimental result is shown in Fig. 8. The formation follows a sinusoidal trajectory in the Y direction and a linear trajectory in the X direction, and the shape parameters are $p = 2$, $q = 4$ and $\beta = 0$, producing a straight line.

B. Lead-follower Implementation

Two different lead-follower configurations are presented. In Fig. 9, a single leader is followed by two robots with cluster shape parameters for distance and heading of $(2, \pi/2)$ and $(4, \pi/2)$ respectively. The two followers move together with some delay with respect to the lead. The formation follows the same sinusoidal trajectory as in the previous example.

In Fig. 10, robot 1 is the leader of robot 2 which in turn is the leader of robot 3. The formation follows again the same trajectory, but in this case a propagation delay can be seen as robot 3 lags with respect to its local leader, which in turn lags with respect to its own leader.

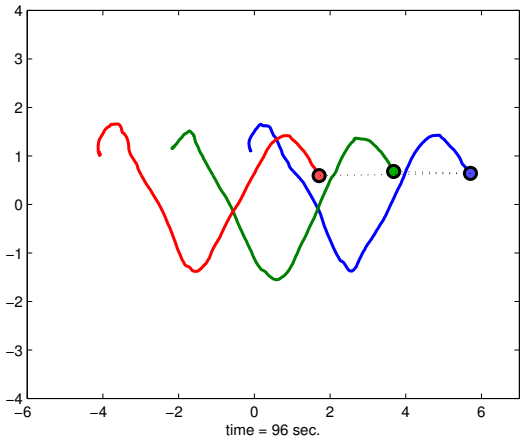

Fig. 8. Formation of a centralized triangle definition. The formation follows a sinusoidal trajectory. The skew angle β is zero to conform to a line.

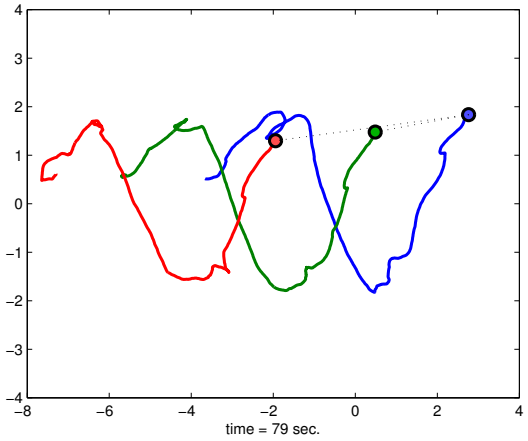

Fig. 9. Formation of a single leader with two followers. The formation follows a sinusoidal trajectory. The lead (blue) follows the trajectory and the followers move together with some lag with respect to the lead.

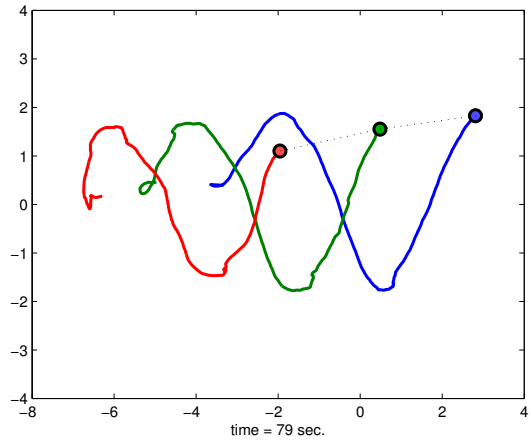

Fig. 10. Formation with local leaders. Robot 1 is leader of Robot 2, which in turn is the leader of Robot 3. The formation follows a sinusoidal trajectory. A lag in the robots motion is propagated along the formation

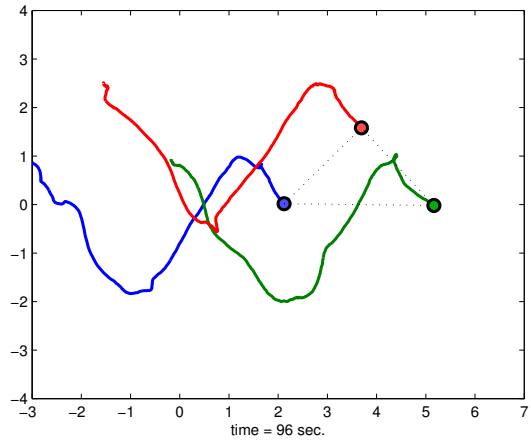

Fig. 11. Triangular formation defined through potential functions. The formation follows a sinusoidal trajectory.

C. Potential Field and Virtual Structure Implementation

The system of Section V is implemented and experimental results are shown in Fig. 11. Desired distances between robots 1 and 2 ($d_{12} \in \mathbb{R}^2$) and robots 1 and 3 ($d_{13} \in \mathbb{R}^2$) are specified. The desired distances between robots define a triangular shape and a goal location -or virtual leader- defines the sine wave trajectory that is followed by the formation.

VIII. Discussion

The selection of cluster space variables dictates the characteristics of the resulting multi-robot system. Definitions where cluster parameters depend on the state of many robot variables tend to favor a centralized implementation approach. This results in the need of global communications which may restrict its use. However, when feasible, this implementation allows for a higher level of abstraction where task-driven definitions make for simple cluster level specifications that result in complex robot level motions. An example of this is the implementation of an escorting/patrolling mission [21] using the centralized definition of Section III.

Low-coupled definitions, like the one presented in Section IV, show low levels of dependencies between variables suggesting a distributed architecture. This allows for low inter-robot communications and shows a higher degree of robustness to robot failures. Also, the system has a level of scalability, since adding new robots to the formation does not affect the definitions of the existing ones. As a drawback, complex formation motions are more difficult to specify compared with centralized definitions. For example, formation rotation in a lead-follower definition implies varying the

bearing angles of all the follower vehicles. Varying only the cluster orientation variable in the centralized case of Section III produces the same behavior. Also, propagation of signals in distributed architectures suffer considerable lag. This can be seen in the experimental results of the in-line formation. In Fig. 8 the centralized approach maintains the desired straight line while in Fig. 9 the two followers lag in their relative position with respect to the leader. Furthermore, in Fig. 10 where robot 1 is the leader of robot 2, which in turn is the local leader of robot 3, the lag increases along the formation chain.

In terms on computational complexity, the decentralized implementations tend to produce sparse Jacobian matrices, lowering the computation burden of the system.

Sections V and VI indicate that the number of cluster variables can vary, resulting in under defined or over defined systems. In the case where the number of cluster parameters are less than the total DOF of the system, unconstrained motions will emerge in the formation. Examples of this are a virtual structure with an undefined orientation [20], or a robot orbiting around its leader [9]. For over defined systems, the operator must make sure that parameter specifications are consistent.

IX. Conclusions

The *cluster space* control approach for planar robots was briefly reviewed for the general case of planar mobile robots. Such framework was then implemented in different ways in order to express some of the most common multi-robot formation control methods from the *cluster space* perspective. First, a centralized definition presented in previous work was reviewed. Then, a distributed architecture of the lead-follower method was presented under the cluster perspective to demonstrate how the same cluster space framework can lead to centralized or distributed architectures depending on the choice of cluster space variables. Potential fields driven formation control and the Virtual Structure method are also considered under our framework perspective. Experimental results using a three-robot system illustrated the different cases and advantages and drawbacks were discussed. The cluster space framework is a general approach to controlling formations of mobile robots and different techniques found in the literature can be interpreted as particular cases within the framework.

References

[1] V. Kumar, N. E. Leonard, and A. S. Morse, *Cooperative Control: A Post-Workshop Volume, 2003 Block Island Workshop on Cooperative Control*. New York: Springer-Verlag, 2005.

[2] Z. Wang and D. Gu, "A local sensor based leader-follower flocking system," *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 3790–3795, May 2008.

[3] R. Fierro, A. Das, J. Spletzer, J. Esposito, V. Kumar, J. Ostrowski, G. Pappas, C. Taylor, Y. Hur, R. Alur, I. Lee, G. Grudic, and B. Southall, "A framework and architecture for multi-robot coordination," *The International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 977–995, Oct-Nov 2002.

[4] A. Das, R. Fierro, V. Kumar, J. Ostrowski, J. Spletzer, and C. Taylor, "A vision-based formation control framework," *Robotics and Automation, IEEE Transactions on*, vol. 18, no. 5, pp. 813–825, Oct 2002.

[5] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, vol. 2, pp. 1217–1222, 2002.

[6] T. Balch and R. Arkin, "Behavior-based formation control for multi-robot teams," *Robotics and Automation, IEEE Transactions on*, vol. 14, no. 6, pp. 926–939, Dec 1998.

[7] F. Schneider and D. Wildermuth, "A potential field based approach to multi robot formation navigation," *Robotics, Intelligent Systems and Signal Processing, 2003. Proceedings. 2003 IEEE International Conference on*, vol. 1, pp. 680–685 vol.1, Oct. 2003.

[8] K.-H. Tan and M. Lewis, "Virtual structures for high-precision cooperative mobile robotic control," *Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, vol. 1, pp. 132–139 vol.1, Nov 1996.

[9] N. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials and coordinated control of groups," *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, vol. 3, pp. 2968–2973 vol.3, 2001.

[10] E. Fiorelli, N. Leonard, P. Bhatta, D. Paley, R. Bachmayer, and D. Fratantoni, "Multi-auv control and adaptive sampling in monterey bay," *Autonomous Underwater Vehicles, 2004 IEEE/OES*, pp. 134–147, June 2004.

[11] C. A. Kitts and I. Mas, "Cluster space specification and control of mobile multirobot systems," *Mechatronics, IEEE/ASME Transactions on*, vol. 14, no. 2, pp. 207–218, April 2009.

[12] I. Mas, O. Petrovic, and C. Kitts, "Cluster space specification and control of a 3-robot mobile system," *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pp. 3763–3768, May 2008.

[13] C. Kitts, K. Stanhouse, and P. Chindaphorn, "Cluster space collision avoidance for mobile two-robot systems," *Intelligent Robots and Systems . IEEE/RSJ International Conference on*, Oct 2009.

[14] P. Mahacek, I. Mas, O. Petrovic, J. Acain, and C. Kitts, "Cluster space control of autonomous surface vessels," *Marine Technology Society Journal*, vol. 43, no. 1, pp. 13–20, 2009.

[15] S. Agnew, "Cluster space control of a 2-robot aerial system," Master's thesis, Santa Clara University, Jun. 2009.

[16] D. Stipanovic, P. Hokayem, M. W. Spong, and D. Siljak, "Cooperative avoidance control for multiagent systems," *Journal of Dynamic Systems, Measurement and Control*, vol. 129, no. 5, pp. 699–707, 2007.

[17] M. Schwager, D. Rus, and J. Slotine, "Decentralized, adaptive coverage control for networked robots," *The International Journal of Robotics Research*, vol. 28, no. 3, pp. 357–375, 2009.

[18] J. Desai, J. Ostrowski, and V. Kumar, "Controlling formations of multiple mobile robots," *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, vol. 4, pp. 2864–2869 vol.4, May 1998.

[19] S. Ge and C.-H. Fua, "Queues and artificial potential trenches for multirobot formations," *Robotics, IEEE Transactions on*, vol. 21, no. 4, pp. 646–656, Aug. 2005.

[20] M. Egerstedt and X. Hu, "Formation constrained multi-agent control," *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 4, pp. 3961–3966 vol.4, 2001.

[21] I. Mas, S. Li, J. Acain, and C. Kitts, "Entrapment/escorting and patrolling missions in multi-robot cluster space control," *Intelligent Robots and Systems. IEEE/RSJ International Conference on*, pp. 5855–5861, Oct 2009.