A Study on Rotational Lock in a Team of Distributed Object Reorienting Robots

Majid Nili Ahmadabadi* **M. R. Barouni Ebrahimi*** **Eiji Nakano****

*Robotics Lab., Dept. of Electrical and Comp. Eng., Faculty of Eng.
University of Tehran, Tehran Iran
E-mail: nily@sofe.ece.ut.ac.ir
**Advanced Robotics Lab., The Graduate School of Info. Sciences
Tohoku University, Sendai Japan
E-mail: majid, nakano@robotics.is.tohoku.ac.jp

Abstract

In this paper, some methods for removing the rotation lock, produced in a distributed multirobot object reorienting system, are studied. The robots implement Constrain and Move strategy [1,2] to turn the object around the desired rotation center. In this strategy, a group of robots constrain the horizontal movements of the object and another team of them pushes the object to turn it. If there is any small position error in the object constraining robots, the rotational movement of the object is locked in one or both directions.

Implementation of some compliant elements in the robot arms to solve the lock problem is considered. Also, due to undesirable effects of the compliant arms on the system performance, a distributed method to remove the lock for stiff robot arms is introduced.

Some simple methods to detect the lock in the system are developed. Also, a series of dynamic simulations are conducted to check the efficiency of implementing the developed strategy on two systems with stiff and compliant arms.

1 Introduction

Simple and low cost cooperative robots can be used to manipulate large or heavy objects [1]. Coordination of the robots and control of the teams in presence of some errors in the robots are among the most challenging problems in such systems.

Leader-follower [3], centralized [4], and distributed architectures [1,5,6,7] are three main coordination styles used in cooperative object manipulating systems. Among these approaches, distributes systems could be made more fault tolerant. Also, in a distributed system, the system performance does not necessarily depend on one of the cooperative agents.

In [1,2] the Constrain-Move strategy for a team of distributed robots to reorient an object around a desired point or to move it on a straight line is introduced. In this strategy, learning from mechanical systems, a group of the robots constrain undesirable movements of the object and another team of the robots pushes the load on the desired path, see figure 1 as an example.

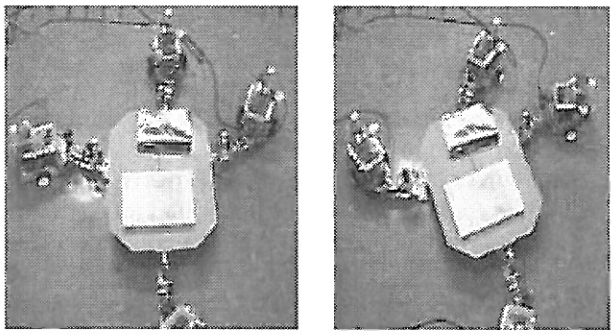

Figure 1: Three robots constrain the object and the robot on the back pushes the load to turn it.

One of the main problems in implementation of this strategy, as in any other cooperative position controlled robots, is that, the object will be locked if there is any error in position of the robots. In [2,7] it is argued that, the problem can be partially solved if some compliant elements are incorporated in the robot arms.

Installation of some compliant units in the robot arms solves the problem when the position errors of the robots are small. Larger errors could be compensated if the robot arms are very soft. But, there will be some undesirable object movements when the robot arms are too soft.

One of the desirable aspects of a distribute system is possessing the ability to detect faults in the system and to solve them automatically in a distributed manner. Therefore, in this paper, a distributed strategy to remove the rotational locks in a system with stiff arms is proposed. This method is also tested on cooperative robots with compliant arms.

The Constrain-Move strategy is briefly introduced in the next section. Then, the reason for formation of the rotational lock and its effects on the system are discussed. A distributed method to solve the problem and a simple measure to detect the lock are introduced in the fourth section. Simulation results are discussed in section 5. A conclusion of this paper is given in the last section.

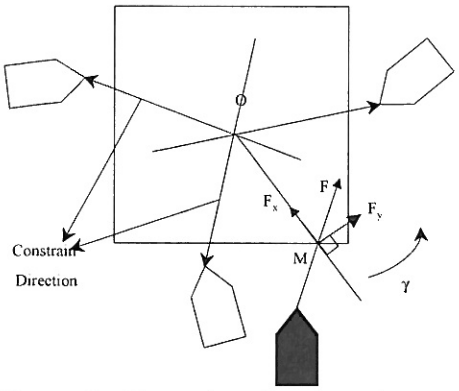

Figure 2: Constraint directions in constraint making robots (white ones).

2 The Constrain-Move strategy for reorientation of an object in place

When moving a train on the rails, the rails interaction forces constrain the train so that it can only move along them. When turning a crank, the reactive wrenches of the ground cancel out all of the external forces but the resulting torque about the crank-ground joint. In these examples, the robot or the operator is not concerned about constraining the object. Therefore, the main job is to control the object along its desired path in such a way that the interaction forces do not exceed their limits or a jamming does not occur. This idea is used to develop the Constrain-Move strategy [1,2].

To turn the object around a point in a plane with Constrain-Move strategy, figure 2, a team of the robots (white ones) must constrain the object in X and Y directions and a robot (the black one) produces an appropriate torque around the object's rotation axis, its free direction. In other words, the robot-object interaction forces in the constraint directions shall make a force direction closure [9] on the object and the object is free to rotate.

2.1 The constraining strategy

As shown in figure2, three robots (white ones) constrain the object's rotation center. To constrain the object, the robots are arranged in a way that their arm directions intersect at the object's desired center of rotation (O) and the angle between each neighboring pairs of arms is less than 180 degrees. Each white robot, called a constrain-making robot, controls its mobile base so that its mobile-arm joint does not go back along the arm direction. Also the robot follows the object in arm-perpendicular direction and, cosequently, the arm-object angle is constant. Doing so, the desired rotation center does not move and the object is free to rotate. The black robot pushes the object with force F. F_y creates a torque around point O and the object rotates.

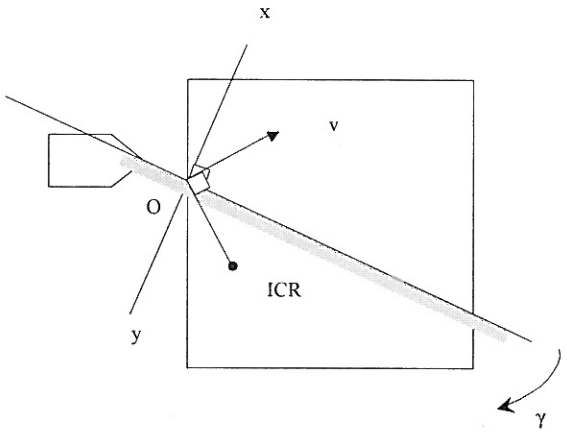

Figure 3: Robot and possible rotation centers of the object in constrain and move strategy.

3 The lock problem

When more than two robots constrain the object, error in position or orientation of the cooperative robot arms may result in a lock on rotation axis of the object, in one or in both directions.

Lets consider one of the object constraining robots, see fig 3. Since the mobile-arm joint does not move backward, the locus of possible rotation center of the object is in the right half-plane of each robot arm, when the object is rotated in γ direction. Therefore, the object instantaneuos center of rotation (ICR) must be inside the area where all of the one-robot-object rotation center locus overlap. We call this area the possible rotation center area (PRCA).

In figure 4, robot 3 is shown in two different situations. Because of having error in orientation of robot 3 arm in both states, the arm directions do not intersect at a common point. In state 1, the locus made for all one-robot-object have no overlap. Therefor, no rotation center for the object can be found when wishing to turn it in γ direction. This is the lock problem and to solve it, the triangle $M_1M_2M_3$ (lock triangle) should be removed.

In state 2 of figure 4, PRCAs of the robots overlap at $M'_1M'_2M_3$. We call this area the common triangle. When turning the object in γ direction, the object rotation center may move inside the common triangle $M'_1M'_2M_3$. If the rotation direction is reversed, the lock triangle $M_1M_2M_3$ is changed to the common triangle and the common triangle $M'_1M'_2M_3$ becomes the lock triangle. This means, having any error in orientation of the constraint making robot arms, the lock problem arises in one direction.

4 Overcoming the lock problem

To solve the lock problem, the lock triangle should be removed. In fact, the lock triangle should be shirked to a point or is changed to a common triangle. In this

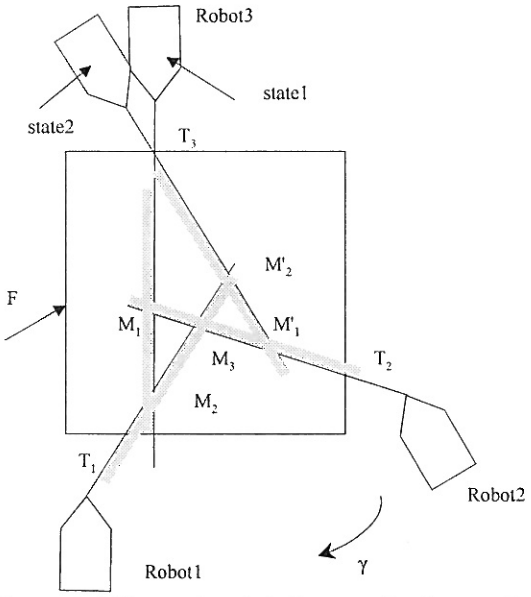

Figure 4: When robot 3 is in state 2, the rotation center is in the common triangle $M'_1M'_2M_3$ and the object can be turned. If robot 3 is in state 1, the object is locked in clokwise direction.

section some methods for solving the lock problem are discussed.

4.1 Compliance and internal forces

Incorporating some compliant elements in the robot arm helps the robot to perform its interactive tasks more easily [2,8]. If the constraint making robots show some compliance to the object movement in their constraint directions, their contact point with the object can have a limited movement in the constraint direction. Hence, the strict condition on possible location of the rotation center for each robot is relaxed. Consequently, the object can be rotated even if there is some error in position or orientation of the robots. But, internal forces are created when using compliant robot arms to solve the lock problem.

In figure 5.a, AB is a constraint making robot arm, point A is the arm-object contact point, and CR is the object center of rotation. Assume that there is no error in the system and CR is on the arm direction. A pure torque turns the object and point A rotates around CR with angular velocity ω (linear velocity v_1) in γ direction, i.e. it moves on arc C. According to our object constraining strategy, the robot follows the object in arm-perpendicular direction to keep its arm-object angle fixed. Therefore, the angle between the robot arm and tangent of arc C (α) is kept constant. As a result, point B (mobile-arm joint) rotates around CR with the same rotation velocity ω. Hence, length of the robot arm spring is constant and the internal forces are zero.

In figure 5.b, there are some errors in the system and CR is not on the arm direction. Assume CR is outside of the robot PRCA. When turning the object, point A

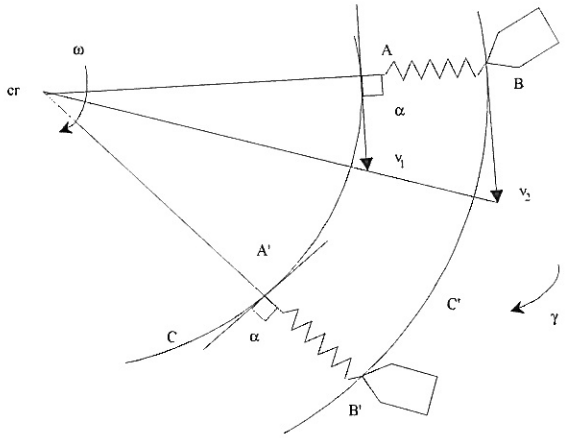

Figure 5.a: If c.r. is along the arm direction, the spring is not compress when the object is turned.

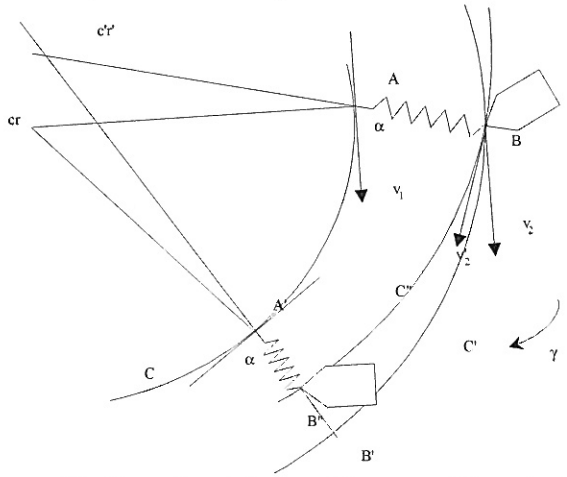

Figure 5.b: If c.r. is not in possible area of the robot, when turnning the object, the spring is compressed.

Figure 5: Internal forces are produced when the object is turned around c.r. and c.r. is outside of possible area of the robot.

has the linear velocity v_1 and point B would have linear velocity v_2, if there were no error in the system. But, since the robot does not move backward along its arm direction, velocity of the mobile-arm joint will be v'_2. Therefore, when A moves on arc C, B will rotate around the virtual center of rotation (C'R') and moves on the C". When A reaches A', B will arrive at B". Consequently, the arm springs are compressed and internal forces are exerted on the object.

4.2 Arm Rotation method

With robot3 in state 1 in fig. 4, assume there are some errors in the system and the lock triangle ($M_1M_2M_3$) is formed. If robot3 arm rotates around T_3 (arm-object contact point) in opposite direction of γ, the lock triangle shrinks. Therefore, after sufficient rotation of robot 3 arm around T_3, going from state1 toward state2, the lock problem in γ direction is solved.

Without loose of generality assume that, only one constraint making robot has error in its arm direction. Three methods to solve the lock problem can be used:

1- The faulty constraint-making robot turns and eliminates the lock triangle (Faulty robot method).

2- One of the non-faulty constraint making robots turns to remove the lock (Non faulty robot method).

3- All constraint making robots turn to eliminate the lock triangle (All robots method).

It seems that, the best results in gained if the faulty robot detects the lock and solves the problem fast. But, the robots cannot easily check which of them has error in its position or orientation. Therefore, it is possible that one of the non-faulty robots tries to overcome the lock problem. We shall check the system performance in our simulations for such situations.

Because of having difficulties in detecting the faulty robot, it is likely that, cooperation of three constraint making robots to eliminate the lock triangle is the most proper solution. Engagement of the faulty robot in the lock elimination is the reason for such expectation.

4.3 A combination of two methods

Very soft robot arms are needed if the lock triangle is large. The main consequences of having very soft arms are larger error in position of the desired rotation center and some vibrations in the system.

On the other hand, for stiff robots, any small error in the constraint making robot arms will cause a rotational lock in the system and the arm rotation method is activated. Therefore, to have a practical system, having compliant robot arms is a necessity. In other words, using the arm rotation method in a system with low compliance object constraining robot arms is proposed. In such systems, small errors in robot arms are compensated and the arm rotation method is used for larger errors.

4.4 Two measures

Two parameters are considered to compare the methods introduced to solve the lock problem.
The first one is displacement of the desired rotation center in the world coordinate system and is defined as:

$$e_r = \sqrt{\Delta x_{d.c.r.}^2 + \Delta y_{d.c.r.}^2} \quad (1)$$

where $\Delta x_{d.c.r.}$ and $\Delta y_{d.c.r.}$ are displacements of the desired rotation center (DRC) in X and Y directions of the world coordinate system.

The second parameter is the internal forces produced due to existence of error in the robots. These forces are a function of the arm stiffness and error in position or orientation of the object constraining robots.
We define F_r as an indirect measure of the internal forces as:

$$F_r = \sqrt{(k_1\Delta x_1)^2 + (k_2\Delta x_2)^2 + (k_3\Delta x_3)^2} \quad (2)$$

where K_i is the stiffness coefficient of the ith object constraining robot arm and Δx_i is its deflection.

4.5 The lock detection

An external observer can easily detect the lock. But, our goal is to make a distributed system with minimum dependency on a central unit and reliance on direct robot communication.

Complete, accurate, and distributed lock detection by each robot is a hard task. Therefore, two primitive, simple, and distributed methods to detect the lock are described. It is assumed that no external disturbances are applied to the object and the object turning robot pushes the object properly.

In the first method, called angular velocity algorithm, $\Delta\theta$ and ω are used where ω is the angular velocity of the object and

$$\Delta\theta = \theta_{des} - \theta_{now} \quad (3)$$

Where θ_{des} and θ_{now} are the desired and the current orientation of the object respectively. After sufficient time from starting the task, the constraint making robots check if ω is in the desired direction and it is small with respect to $\Delta\theta$. This condition means that the object is not rotating as expected and a lock is occurred. Therefore, one-step arm rotation is performed. After a delay, the robots check the condition again. This loop will be continued until the lock is removed.

Late detection of the lock in the beginning of the task is the main drawback of this method. This delay may cause production of big internal forces. Requirement for setting a proper relation between $\Delta\theta$ and ω for each system is another restriction on this algorithm.

The pushing forces in the robot arms are increased when a lock is occurred. Therefore, these forces (arm spring compressions) are used in the second method (internal force algorithm) for the lock detection. In this method, if deflection in a robot arm is more than a defined value, the robot assumes that a lock is occurred.

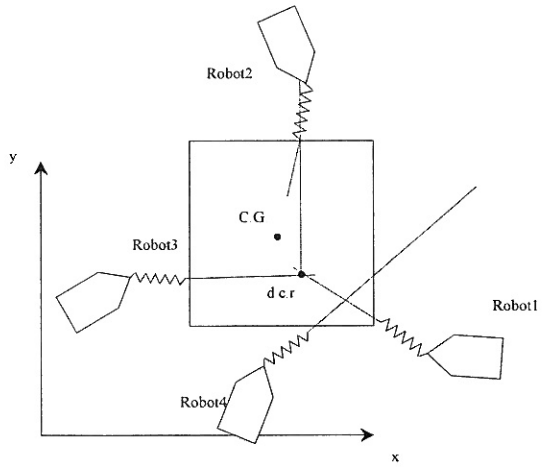

Figure 6: Geometrical model of the simulated system

The main problem of this algorithm is that, the robots do not consider $\Delta\theta$, while the object turning robot adjusts its pushing force with respect to $\Delta\theta$. Therefore, the arm spring deflections may be small when the lock is occurred. Begin disable to distinguish the external disturbances from the robot forces is another drawback of this method.

If we assume that there is no external force on the system, a combination of the mentioned methods could be used. In other words, if the arm spring deflections are large, the second method detects the lock. The first method finds the lock when the relation of ω and $\Delta\theta$ is different from what is expected. Doing so, the lock in the begining of the task and near the goal configuration can be detected.

It is noteworthy that, these methods are not robust enough to be used in a real system. More studies are needed to find better and distributed lock detection algorithms.

5 Simulation results

A series of dynamic computer simulations are conducted to study effects of the arm rotation algorithm and incorporation of compliant units in the robot arms on the system performance.

Figure 6 shows the model used in the simulations. In this model, the object is a 0.5×0.5 square of unit mass and moment of inertia. The constraint making robots (robots 1, 2, and 3) are located at (0.25, -0.2), (0.1,0.25), and (-0.25, -0.05) respectively. It is assumed that there is 6 degrees error in orientation of robot 2 arm. The maximum force applied by the object turning robot, robot 4, is set to 4N. The goal orientation of the object is set to 30 degrees. Each object constraining robot assumes the lock is occurred when its pushing force is more than 10N or $|100\Delta\theta - 250\omega| > 150|\omega|$.

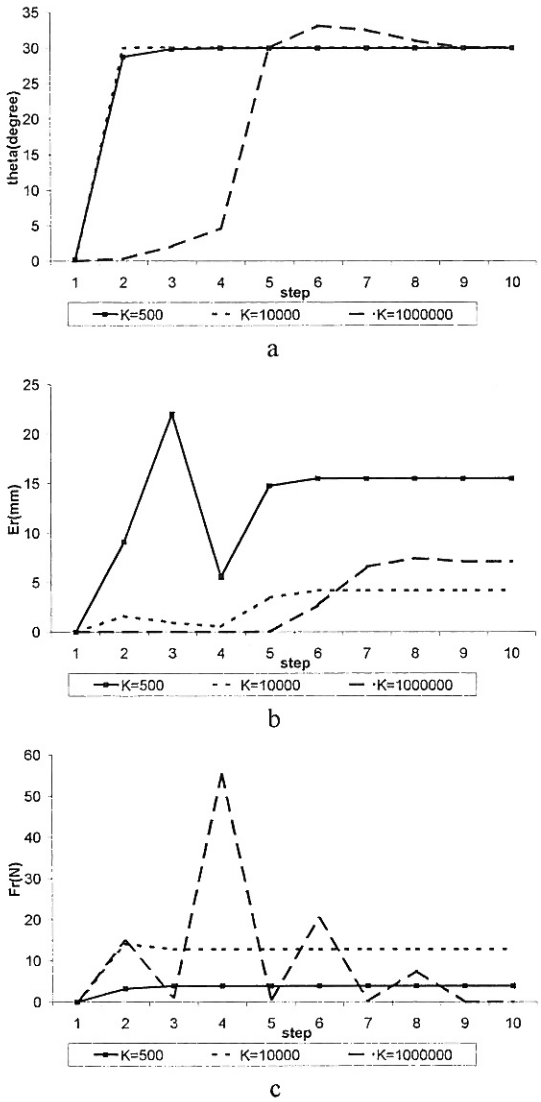

Figure 7: Object angle, e_r, and F_r for different values of K when all robots try to eliminate the lock.

For all stiffness coefficients (K) used in this simulations, the robots could not turn the object without executing the arm rotation method. In the results shown in figure 7, all robots try to remove the lock. The highest position error and the lowest internal force are gained for the most compliant arm (K=500).

It can be observed that, for K=1,000,000 e_r is larger compared to the case where K=10,000. The reason for getting such results is that, the forces are higher for the stiffer robots and the robots try to eliminate the lock many times. Therefore, the common area is formed and a larger e_r is gained.

In figure 8, K is equal to 10,000 and three arm rotation methods (faulty, non-faulty, and all) are used. In this simulation, the arm rotation angle in each step is smaller than in the above mentioned simulations (Fig. 7); i.e. lock removal is slower. The results show that, we get minimun e_r if the faulty robot tries to eliminate the lock triangle. Formation of the common area in two other metods is the reason of getting a larger e_r.

The lock is removed faster when all robots try to eliminate it. Therefore, minimum internal forces are produced in this case, see figure 8 graph c.

A comparison of the results for K=10,000 in figure 7 and all robots method in figure 8 shows that, e_r is smaller when the lock is removed faster. The reason for getting such results is that, the object's center of rotation moves toward the desired rotation center faster when the lock removal procedure acts more quickly.

6 Final remarks

In this paper, the reason for formation of the rotational lock in a team of distributed object turning robots is studied. Incorporation of compliant elements in the robot arms to prevent the system from locking is considered first. Then, to reduce the problems caused by the soft robots, a distributed method to remove the lock is introduced. Also, simple procedures to detect the lock in the system are developed.

The simulation results show that, our method gives the best results when there is a reasonable compliance in the robot arms. Also, the speed of the lock detection and removal are two main parameters affecting the system performance.

More effective and advanced distributed methods to detect the lock must be studied. Enhancement of the strategy to solve the lock problem for a higher number of object constraining robots is another direction of our current research.

7 References

[1] Majid Nili Ahmadabadi and Nakano Eiji "A Constrain and Move approach to distributed object manipulation", Accepted in the IEEE Transaction on Robotics and Automation.

[2] Majid Nili Ahmadabadi and Nakano Eiji "Constrain and Move: A new concept to develop distributed object transferring protocols" In Proc. of The 1997 IEEE Int. Conf. On Robotics and Automation (ICRA '97), pp. 2318-23 25, New Mexico-USA

[3] Nakano E. et. al, " Cooperational control of the anthropomorphous manipulator MELARM, " Proc. of 4th Int. Symp. on Industrial Robots, pp. 251-260,1974

[4] Y. Nakamura "Advanced robotics: Redundancy and optimization", Addison-Wesley, 1990

[5] Z.D. Wang, Eiji Nakano, and Takuji Matsukawa "Realizing cooperative object manipulation using multiple behavior-based robots", In Proc. of the 1996 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems(IROS96), Vol. 1, pp. 310-317 Osaka-Japan, 1996

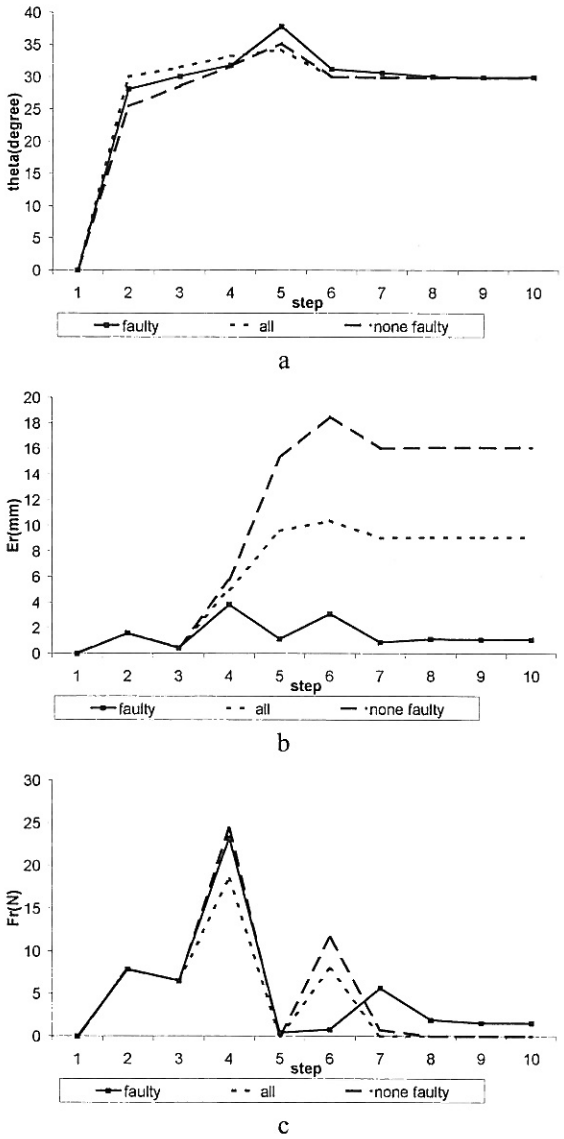

Figure 8: Object angle, e_r, and F_r, when K=10,000 and speed of lock removal process is low.

[6] Donald B.R., Rus D., and Jenning J., "Information invariants for distributed manipulation", Int. Journal of Robotics Research, Vol. 16, No. 5, pp. 673-702, October 1997

[7] Majid Nili Ahmadabadi and Nakano Eiji. "Task allocation of object transferring robots" In Proc. of the 1997 IEEE/RSJ Int. Conf. On Intelligent Robots and Systems (IROS '97) Vol., pp. 435-440, Grenoble-France

[8] Piero Velletri "Study of a robot manipulator incorporating passive compliance" Master thesis, Graduate School of Info. Sciences, Tohoku University, 1996

[9] Y. Yoshikawa "Passive and active closures by constraining mechanisms", In Proc. of the 1996 IEEE Int. Conf. Robotics and Automation (ICRA'96), pp. 1477-1484, 1996