2014 IEEE International Conference on
Automation Science and Engineering (CASE)
Taipei, Taiwan, August 18-22, 2014

Advanced-Collision-Map-Based On-line Collision and Deadlock Avoidance between Two Robot Manipulators with PTP Commands

Ahmad Yasser Afaghani and Yasumichi Aiyama, *Member*, *IEEE*

Abstract— **This research aims to build an on-line system for avoiding collisions and deadlocks between two robot manipulators which are controlled using point-to-point (PTP) commands. Both robots are sharing the same workspace and have no prior knowledge of the commands which will be sent after starting the system.**

We have proposed a collision map concept for detecting the collisions between the end-effectors of the robots [1]. In this work, an advanced collision map method has been created for detecting the collisions between the whole body of the robots and representing them as collision areas on the map. Moreover, the map has been used to avoid the deadlocks which can be occurred if any robot becomes an obstacle in front of the other. To realize a collision-free trajectory of the robots, time scheduling of command execution time has been applied to avoid any collision areas on the map. The system has been tested on an openGL-based simulator for demonstrating the effectiveness of the system.

I. Introduction

Since many industrial applications include multiple robot manipulators to achieve several tasks in the same workspace, the collision avoidance becomes a significant matter to be considered. Off-line motion planning [2] is the common method which is widely used in FA. But when the control commands of each robot are unpredictable due to unstructured environment such as bin picking, it is necessary to consider on-line motion planning [3]. This paper highlights on the second kind of motion planning which are still under development.

In the previous works, the collision avoidance has been solved using abundant concepts [4]–[10]. On the other hand, many methods have been developed for on-line collision avoidance. Khatib [11] presented a unique method for obstacle avoidance by manipulators and mobile robots using artificial potential field concept. Kalaycioglu *et al.* [12] suggested a modified impedance control concept for solving the collision problem. Cheng [13] presented a new approach which utilizes planned collision-free paths for independent tasks using a 2D geometric model in consideration of the swept region by the robot arms. Cellier *et al.* [14] proposed the reflex action theory. For example, when the second arm happens to collide with the first arm, the shape of the protection zones is modified. Freund *et al.* [15] presented a new method CARE (Collision Avoidance in Real-time Environment) to avoid collisions by changing the predetermined path for endangered robots using the redundant kinematics of a robot. Bosscher *et al.* [16] proposed an algorithm for collision avoidance by creating a set of linear inequality constraints on the commanded velocity of the robot by formulating joint limits and potential collisions. In tele-operated robots, e.g. in medical and military applications, on-line collision avoidance has been considered and studied extensively as well [17] [18].

Meanwhile, point-to-point (PTP) command-based control is widely used for industrial robots. These commands are sent from the application module which is responsible for distributing the commands to each robot based on the tasks. However, the controller of industrial robots is a black box i.e. no modification is possible in the controller system. Thus, the collision avoidance for such robots in on-line mode becomes a significant issue. There are a few number of research papers have tackled that issue. Zhou *et al.* [19] proposed a zone-blocking method. When one robot enters into the common workspace, a flag is activated to prevent other robots from entering the workspace. The method is simple and safe but not very efficient.

This work proposes an efficient method for on-line collision avoidance between two robot manipulators considering the whole body as an extension of [1]. The robots are controlled using unpredictable PTP commands which are sent after starting the system. Therefore, the collision map method, which has been proposed for detecting collisions between only end-effectors (EEFs) of two robot arms, has been ameliorated in order to include the collision information between all links of the robot body. Furthermore, in some cases and due to unpredictable control commands, both robots may reach to a deadlock situation i.e. both robots become obstacles to each other. Consequently, the new map has been used intelligently to avoid the deadlocks.

II. System description

The control restrictions of industrial robot systems have considered as conditions to design the collision avoidance system. 1) When the robot executes a PTP command, the motion of the robot becomes restricted and unchangeable until reaching the target. 2) The path and trajectory of the PTP command which is sent from application module are unmodifiable. Thus, they are regarded as high priority in collision avoidance process.

To solve the problem of collision avoidance, we proposes to include on-line planner between application and controller modules. Thus, the entire system structure is composed of three substantial modules. The first module is *Application Module*, which provides PTP commands to both robots R1

A. Y. Afaghani and Y. Aiyama are with Manipulation System Laboratory, University of Tsukuba, Tsukuba, Ibaraki, Japan. yasser@ms.esys.tsukuba.ac.jp, aiyama@esys.tsukuba.ac.jp.

978-1-4799-5283-0/14/$31.00 ©2014 IEEE

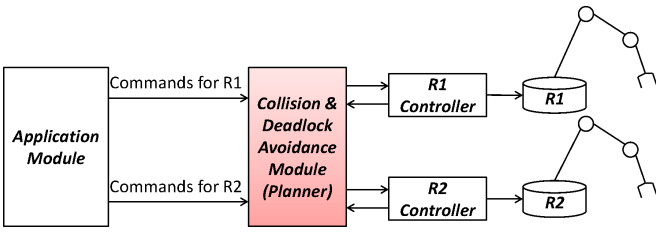


Fig. 1. Overview of collision avoidance system.

and R2 in sequence. Then these commands are acquired by the *Collision & Deadlock Avoidance Module* one after the other. This module is the planner which is responsible for detecting collisions between the two arms on the basis of received commands as well as the current data acquired from the robots. Subsequently, the scheduling of the commands' execution timing for avoiding any collisions and deadlocks is performed before sending the commands to the robot. Finally, the *Robot Controller Modules* are responsible for controlling the robots and providing the planner with the necessary information from both robots such as posture, speed, and acceleration in order to perform collision detection. The system modules are illustrated in Fig. 1. This system has been built assuming that there are no obstacles in the workspace and the paths of the robots are straight lines.

III. Collision Detection

The workspace is equipped with two robots, namely, $R1$ and $R2$. The collision detection between these robots is performed each time the planner acquires a new PTP command for one of these robots.

A. Modes of Robot Manipulator

The robot modes are defined under the system conditions for the purpose of designing the map. We denoted to the robot which has executed PTP commands and committed to its path as *Moving Robot* (MR). On the other hand, the robot is denoted as *Standby Robot* (SR) as long as it is not moving i.e. the robot is waiting for new command or the robot has acquired it.

Here, three different combinations of robot modes can be obtained, MR-MR, SR-MR, and SR-SR. In MR-MR combination, no collision detection is needed because the algorithm is designed to execute PTP command for any robot after assuring that its path is free of collision.

B. Modelling of Robot Manipulator

Modelling the robot manipulator plays an important role in detecting collisions between robots. The model should be precise while being based on a simple mathematical representation for preventing high calculation cost. Many researchers have suggested different modelling techniques e.g. spherical, cylindrical, and polyhedral modelling. In addition to some other modelling using bounding box volume such as discrete-orientation polytopes and oriented bounding box.

In order for the modelling to be functional in an on-line system, the *Swept Sphere Volume* (SSV) modelling which integrates tightness and mathematical simplicity is proposed. SSV is a volume formed by moving a sphere with certain radius on a specified primitive such as a point, line, or rectangle. By changing the primitive dimensions and radius of the sphere, it is possible to model any robot link as tight as possible. The SSV modelling method has an advantage of easiness of collision detection due to the rotational invariant characteristic of the sphere. This facilitates finding of the shortest distance between the primitives and compares the it with radii summation of the spheres which sweep on the primitives. Thus, the computational cost is low for this modelling.

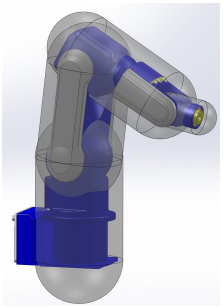


Fig. 2. Modelling robot arm using swept sphere volume.

Since many industrial robot manipulators have a near-tube shape, a line primitive is utilized for modelling the whole body of the robots. This research uses two robots, each one is constructed of a base, two links, and an end-effector (EEF). Therefore, each robot is modelled using 4 line segments with appropriate radii of swept spheres. The robots' CAD design is shown in Fig. 2.

C. Design of Advanced Collision Map

Let start to address the collision detection in the case of SR-MR combination at first. Assume that a new PTP command has been acquired by the planner to move $R2$'s EEF from $S2$ to $E2$ with path length l_{SR}, and requisite travelling time t_{tar}^{SR}. At this moment, robot $R1$ has already adhered to its path so it is MR where $R1$'s EEF is located at $P1$ and heading toward $E1$. The necessary time to reach the target from $P1$ is t_{tar}^{MR}. A simplified sketch of the workspace with two robots and paths are shown in Fig. 3.

Let us denote the SSV line segments which model $R1$ and $R2$ by seg_j^{MR} and seg_i^{SR} respectively. Where, $i \in [1, m]$ and $j \in [1, n]$ are indicated by segment number,

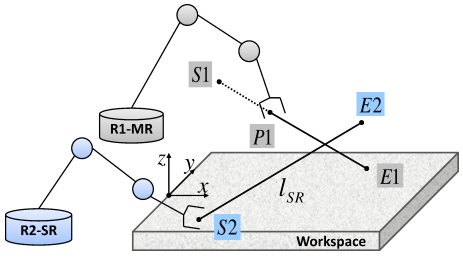


Fig. 3. Paths of robots in shared workspace.

TABLE I

RANGES OF COLLISION TIMINGS OBTAINED BETWEEN EACH PAIR OF SEGMENTS OF MR AND SR.

	seg_1^{MR} (Base)	seg_2^{MR} (Link1)	$\cdots$	seg_n^{MR} (EEF)
seg_1^{SR} (Base)	$T_{11}(t,k)$	$T_{12}(t,k)$	$\cdots$	$T_{1n}(t,k)$
seg_2^{SR} (Link1)	$T_{21}(t,k)$	$T_{22}(t,k)$	$\cdots$	$T_{2n}(t,k)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
seg_m^{SR} (EEF)	$T_{m1}(t,k)$	$T_{m2}(t,k)$	$\cdots$	$T_{mn}(t,k)$

and n and m are the maximum number of segments for the robots. In order to detect collisions between a pair of segments, first, a minimum distance between both segments are calculated $d_{min}(seg_i^{SR}, seg_j^{MR})$. Then, the minimum distance is compared with the summation of swept spheres radii of both segments. If that distance is equal or less than the result of the summation, then, there is a collision between two segments.

$$d_{min}(seg_i^{SR}, seg_j^{MR}) \leq r_i + r_j \tag{1}$$

Where, r_i and r_j are radii of spheres which sweep on seg_i^{SR} and seg_j^{MR} respectively.

Here, we define a new collision mapping concept. It relates the servo time of MR with travelling time of SR. It assumes that SR starts to move as soon as the planner acquires the command, so, SR will move from initial to target position within a time period of t_{tar}^{SR}. To design the aforementioned map, a collision is tested every sampling time dt_s along servo time of MR $t \in [0, t_{tar}^{MR}]$. So, at time t, MR's segment seg_j^{MR} has a specific posture which can be acquired by inverse kinematic. That segment posture is checked with SR's segment seg_i^{SR} for all postures every time sample along $\tau \in [0, t_{tar}^{SR}]$. As a result, ranges of collision timings are obtained $T_{ij}(t,1), T_{ij}(t,2) ... T_{ij}(t, rng_{ij}(t))$. Where, $rng_{ij}(t)$ is the maximum number of ranges at time t for a checked pair (ij). The range at time t for pair (ij) is referred to as a continuous collision and starts from $\tau_{ij}^{str}(t,k)$ and ends at $\tau_{ij}^{end}(t,k)$ whereas $k \in [1, rng_{ij}(t)]$ refers to the range number. After finishing all collision checks for all pairs of segments, $m \times n$ results of collision ranges are acquired as listed in Table I. An illustrative graph of the new collision mapping of pair (ij) is shown in Fig. 4. Afterwards, for determining the potential collisions between the whole bodies of both robots, a union of the ranges every time sample within $t \in [0, t_{tar}^{MR}]$ for all pairs is calculated as follows:

$$\bigcup_{i=0}^{i=m} \bigcup_{j=0}^{j=n} \bigcup_{k=0}^{k=rng_{ij}(t)} T_{ij}(t,k) \longrightarrow \begin{bmatrix} T_{Total}(t,1) \\ T_{Total}(t,2) \\ \vdots \\ T_{Total}(t, rng_{Total}(t)) \end{bmatrix} \tag{2}$$

The result is a group of total ranges of collision timings $T_{Total}(t,1), T_{Total}(t,2) ... T_{Total}(t, rng_{Total}(t))$. Where, $rng_{Total}(t)$ is the maximum number of total ranges at time t, and each total range also has a boundary which starts from $\tau_{Total}^{str}(t, k_{Total})$ and ends at $\tau_{Total}^{end}(t, k_{Total})$ whereas $k_{Total} \in [1, rng_{Total}(t)]$ refers to the total range number.

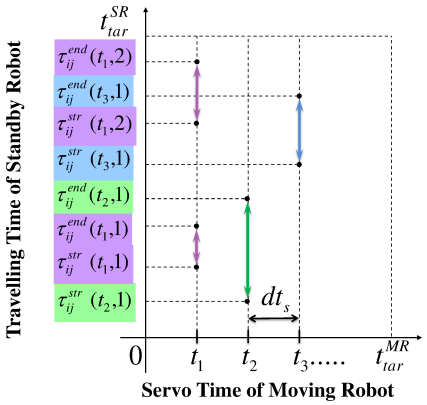


Fig. 4. Map for presenting the ranges of collision timings between segment j of MR and segment i of SR (Best viewed in color).

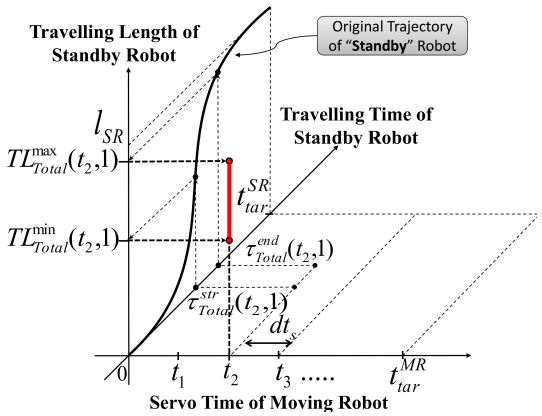


Fig. 5. The method of changing from time space to travelling length space using trajectory information of standby robot EEF.

Based on the original collision map scheme which relates the travelling length of EEF corresponding to the servo time, it is necessary to covert the ranges aforementioned to ranges of collision lengths, i.e. converting from travelling time space to travelling length space. For this purpose, trajectory information of the SR is used. At every time sample along $t \in [0, t_{tar}^{MR}]$, the range $T_{Total}(t, k_{Total})$ is used to get the travelling length of EEF's trajectory only at start and end time $\tau_{Total}^{str}(t, k_{Total})$ and $\tau_{Total}^{end}(t, k_{Total})$ respectively:

$$TL_{Total}^{min}(t, k_{Total}) = f_{TL}(\tau_{Total}^{str}(t, k_{Total})) \tag{3}$$

$$TL_{Total}^{max}(t, k_{Total}) = f_{TL}(\tau_{Total}^{end}(t, k_{Total})) \tag{4}$$

Where, $TL_{Total}^{min}(t, k_{Total}), TL_{Total}^{max}(t, k_{Total})$ are the minimum and maximum travelling lengths of the range at time t, and f_{TL} is the function for calculating the travelling length of SR at a desired time assuming that $f_{TL}(0) = 0$. A graph which relates SR's travelling length against to the servo time of MR by means of travelling time of SR is illustrated in Fig. 5. It is shown how to obtain a range of collision lengths at time t, where, SR's trajectory is shown as a black curved

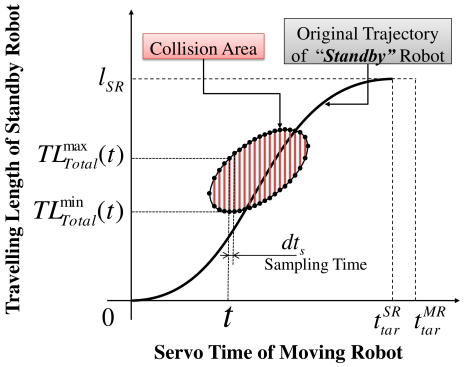


Fig. 6. Collision map shown collision area with original trajectory of standby robot assuming that $t^{MR}_{tar} > t^{SR}_{tar}$.

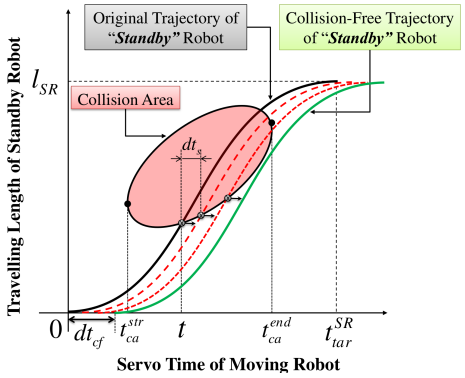


Fig. 7. Description of calculating necessary delay time to avoid collision area.

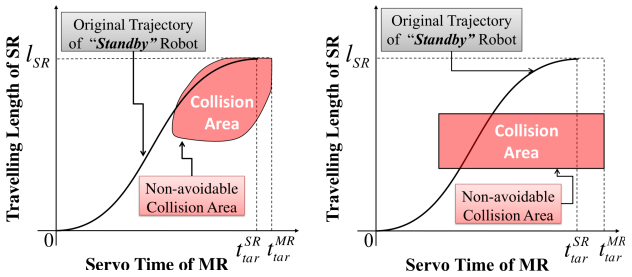


Fig. 8. Illustrative examples of non-avoidable collision areas.

line and the range with a red straight line. By drawing all the ranges on that graph, an area called the *Collision Area* will be formed as shown in Fig. 6. This figure is a graph of the final *Advanced Collision Map* which relates to SR's travelling length with the servo time of MR. The final arrival time of MR t^{MR}_{tar} might be smaller, equal to, or larger than the final arrival time of SR t^{SR}_{tar}. Here, the graph depicts the original trajectory of SR passing through the collision area assuming that $t^{MR}_{tar} > t^{SR}_{tar}$.

In the SR-SR combination, if the planner acquires a PTP command for one of the robots; impose it is $R1$, then, the other robot $R2$ becomes a static object in front of $R1$. Thus, the detection is only performed between the current posture of $R2$ and all postures of $R1$ for the entire travelling time samples under the original trajectory of $R1$. Here, the algorithm is sufficed to detect at least one collision between any pair of segments at any time to judge that there is an *Inevitable Collision* which cannot be avoided i.e. the newly acquired command can not be executed.

IV. Time Scheduling for Collision Avoidance

The condition of this work, as mentioned in (Section II), is to deal with the PTP commands which are sent by application module as they are. This means no change is allowed to the geometric path which are given by the PTP command. Therefore, time scheduling method has been applied. This method modifies the command's execution time while fixing the geometric path.

In the SR-MR combination, for avoiding the collision area on an advanced collision map, the original trajectory of SR is shifted by a necessary delay time required to produce a collision-free path of SR. This delay time is calculated using Algorithm 1. As a result, a final collision-free trajectory of SR is calculated. This trajectory is shown by a curved line in the rightmost of Fig. 7.

The output of the aforementioned algorithm is a delay time dt_{cf} which has three possible cases:
1) $dt_{cf} = 0$: means that there is no potential collision between the robots.
2) $dt_{cf} > 0$: means that there is a potential collision, and the time of command execution should be postponed.
3) $dt_{cf} = \infty$: means there will be an inevitable collision if the command is executed directly.
The third case may happen if the original trajectory of SR passes through the collision area which has an end time $t^{end}_{ca} = t^{MR}_{tar}$ i.e. the collision still exists even after the end travelling of MR. We refer to this area as *Non-Avoidable Collision Area*. Consequently, SR will be prevented to move, so, we get a *deadlock* situation which will be discussed in the next section. The deadlock may happen in the SR-SR combination as well, if the collision is detected between both robots. Some examples for non-avoidable collision areas are shown in Fig. 8.

Algorithm 1 Calculating the delay time dt_{cf} to produce collision-free trajectory

Require: Start and end timing of Collision area t^{str}_{ca} and t^{end}_{ca}

Initializing: $dt_{cf} = 0$;$collision = true$
while $(collision)$ **do**
  for $(t = t^{str}_{ca}$ to $t^{end}_{ca}])$ **do**
    $collision \leftarrow false$; $TL \leftarrow f_{TL}(t - dt_{cf})$
    if ($TL^{max}_{Total}(t, k_{Total}) \geq TL \geq TL^{min}_{Total}(t, k_{Total})$) **then**
      Increase dt_{cf} by one sampling time $1 \times dt_s$
      $collision \leftarrow true$
      break
    end if
  end for
end while

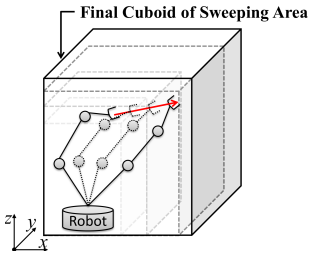


Fig. 9. Approximated sweeping area of robot motion using cuboid

Since the collision map is built while the system is operating, it is necessary to consider the time of building the map dt_m. This time is used to calculate the final delay time for obtaining collision-free trajectory of SR. To this end, a software timer has been added. Thus, the final delay time required to avoid the collision area in on-line mode is: $dt = dt_{cf} - dt_m$

V. Deadlock Avoidance

The deadlock situation is common problem in on-line systems. That because the acquired PTP commands are unknown of timing and target. Thus, the motion of the robots can not be planned before operating the system. In this study, the deadlock can be occurred in two cases:

1) SR-MR combination: When the output of collision avoidance algorithm is $dt_{cf} = \infty$, SR will wait MR to accomplish the command and turn into SR mode. Then, this robot becomes obstacle in front of the other.
2) SR-SR combination: If the result of collision detection is inevitable collision. Here, the first robot which acquired a PTP command is treated as high priority robot and the other robot considered as obstacle.

Where, the robot which becomes obstacle is denoted as SR_{obs}. In general, a new combination is acquired, SR-SR_{obs}. The deadlock avoidance can be performed through finding a safe position, then, command SR_{obs} to move toward the new position. We refer to this position as *Escaping Position*. In this scope, two indispensable information should be decided for calculating the escaping position, the direction of escaping and the distance needed to escape. The concept is going to be explained assuming that SR robot has a PTP

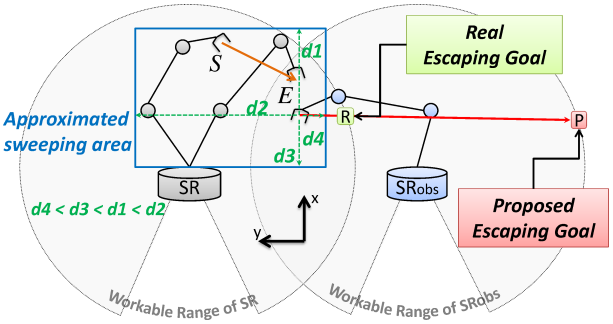


Fig. 10. Description of deadlock avoidance concept using 2D-space

command to move from S to E with path length l_{SR} and the time needed to reach the target is t_{tar}^{SR}.

A. Direction of Escaping

The direction of deadlock avoidance is decided based on approximated sweeping area of SR and current position of EEF of SR_{obs}. The sweeping area of SR is an area which the robot occupies during the motion from initial to target position. This area is approximated by a cuboid. The initial form of the cuboid is fitting the initial posture of SR. This cuboid is re-formed at each time sample along t_{tar}^{SR} based on the calculation result of SR's posture at that time. In other words, the cuboid is expanding at each time a joint of the robot exceed the borders of the cuboid. Thus, at the end of travelling time, a final cuboid is calculated as shown in Fig. 9. The purpose of forming the cuboid is informing the other robot about the danger area which should be avoided. Then, for deciding the direction, a distance between the EEF of SR_{obs} and each side of the cuboid is calculated. As a result, 6 possible directions for escaping can be proposed, X, -X, Y, -Y, Z and -Z. These directions are arranged in ascending order regarding the distances. Thus, the direction with minimum distance will be suggested to be the direction of escaping at first. If this direction is not free of collision, the next direction will be chosen and so on.

B. Distance of Escaping

The distance of escaping can be obtained using advanced collision map. For simplicity, we are going to explain the concept using 2D-space. Assuming SR will move from S to E. First, the direction of escaping is decided based on sweeping area, then, a proposed goal for escaping is set to be a position on the boundary of the workable area of SR_{obs} in the direction of escaping (P). An illustrative figure of deciding escaping position is shown in Fig. 10.

Since, both robots are in standby mode, thus, the trajectory of each robot is able to be scheduled. But initially, the advanced collision map is created considering that the trajectory of SR_{obs} is modifiable, while SR's one is fixed. As mentioned before, the deadlock is caused by SR_{obs} because it becomes as an obstacle, therefore, there is inevitable collision if it did not move. Thus, by creating the advanced collision map for SR_{obs} with new proposed goal, an collision area is formed in the lower right part of the map. As a result, three possible areas on the map can be obtained:

1) The trajectory of SR_{obs} does not pass through the collision area as shown in Fig. 11(a). In this case, we will try to find a shorter distance using the characteristic of the area in order to prevent long travelling of escaping. Hence, the maximum height of the area h_{max} is calculated. This indicates the maximum travelling length among all collision ranges of the area. Since the trajectory of SR_{obs} is safe after that travelling length. Thus, the trajectory is modified to start deceleration at t^* whose travelling length is h_{max}. As a result, the modified trajectory will determine final distance of escaping i.e. the final goal for avoiding the deadlock. We refer to this goal as *Real Escaping Gaol*.

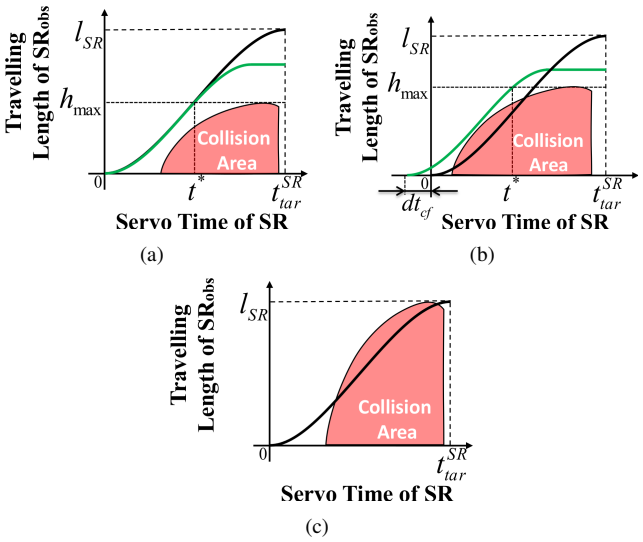


Fig. 11. Possible collision areas in case of deadlock avoidance.

2) The trajectory of SR_{obs} passes through the collision area which has maximum height $h_{max} < l_{SR}$ as shown in Fig. 11(b). Here, the area can be avoided if the trajectory is shifted back to $t < 0$ which means executing SR_{obs} before SR with necessary delay time dt_{cf}. Furthermore, the trajectory of SR_{obs} can be modified based on the maximum height of the area as mentioned in the first case.
3) The maximum height of collision area $h_{max} = l_{SR}$ as shown in Fig. 11(c). This means there is inevitable collision, so, the direction of escaping should be changed to the next proposed one which is mentioned in previous subsection.

VI. Simulation and Evaluation

The collision avoidance system has been tested and evaluated on two robots using an openGL-based simulator. In this work, two Yaskawa motoman robots (HP3J and UPJ) have been imitated using this simulator. The robots are fixed on a board, with a distance between the central axis of the robots' bases being $500mm$. The global coordinate system is located in the middle between both robot bases. A snapshot of the simulator is shown in Fig. 12.

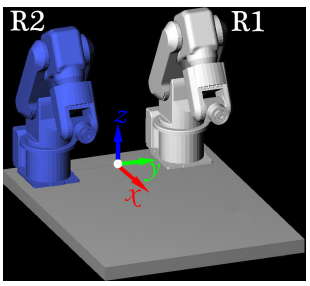
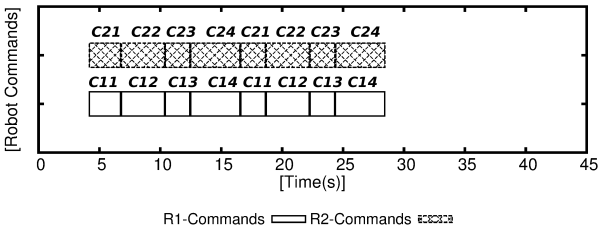


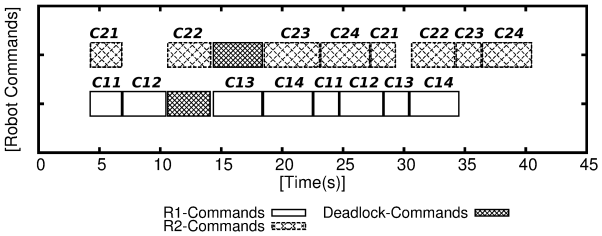
Fig. 12. Snapshot of Open-GL based simulator which illustrates the two robots in initial posture(best viewed in color).

TABLE II
PTP commands which are used to control R1 and R2.

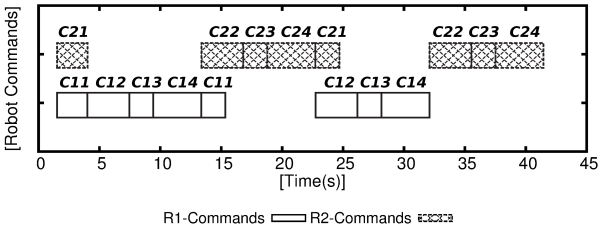
$R1$-**Commands (x,y,z,roll,pitch,yaw)**	$R2$-**Commands (x,y,z,roll,pitch,yaw)**
$C11(300, 250, 330, 0, 0, 0)$	$C21(300, -250, 330, 0, 0, 0)$
$C12(350, 0, 200, 0, 0, 0)$	$C22(350, 0, 200, 0, 0, 0)$
$C13(350, -51, 200, 0, 0, 0)$	$C23(350, 51, 200, 0, 0, 0)$
$C14(400, 250, 330, 0, 0, 0)$	$C24(400, -250, 330, 0, 0, 0)$



(a) Command chart without collision avoidance system.



(b) Command chart of applying new proposed method (Advanced collision map).



(c) Command chart of applying previous method (Zone-blocking [19]).

Fig. 13. Commands operation times of R1 and R2.

A. Simulation Results

Many examples have been examined using proposed collision avoidance system. In this paper, one of these examples which contains collisions and deadlocks is presented. The example is used PTP commands which are illustrated in the TABLE II. Each command includes information of EEF posture according to the global coordinate system $[x(mm), y(mm), z(mm), roll(deg), pitch(deg), yaw(deg)]$. The experiment is divided into two parts. The first part is performed without collision & deadlock avoidance module, i.e. the commands are sent directly to the robot controller. The second part is carried out including collision & deadlock avoidance module. Let us assume that $R1$ is the gray color robot and $R2$ is the blue robot, and

TABLE III

TRACKING RESULTS OF MINIMUM DISTANCE BETWEEN THE LINKS OF BOTH ROBOTS WITHOUT APPLYING COLLISION AVOIDANCE SYSTEM

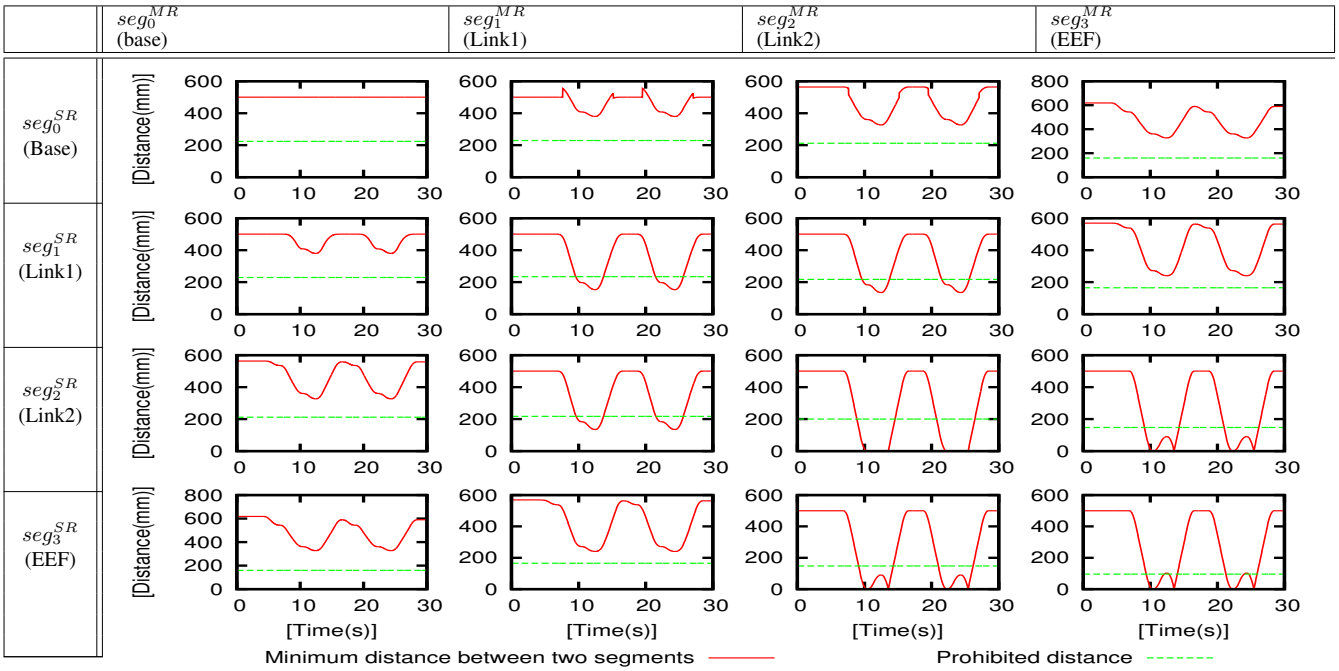
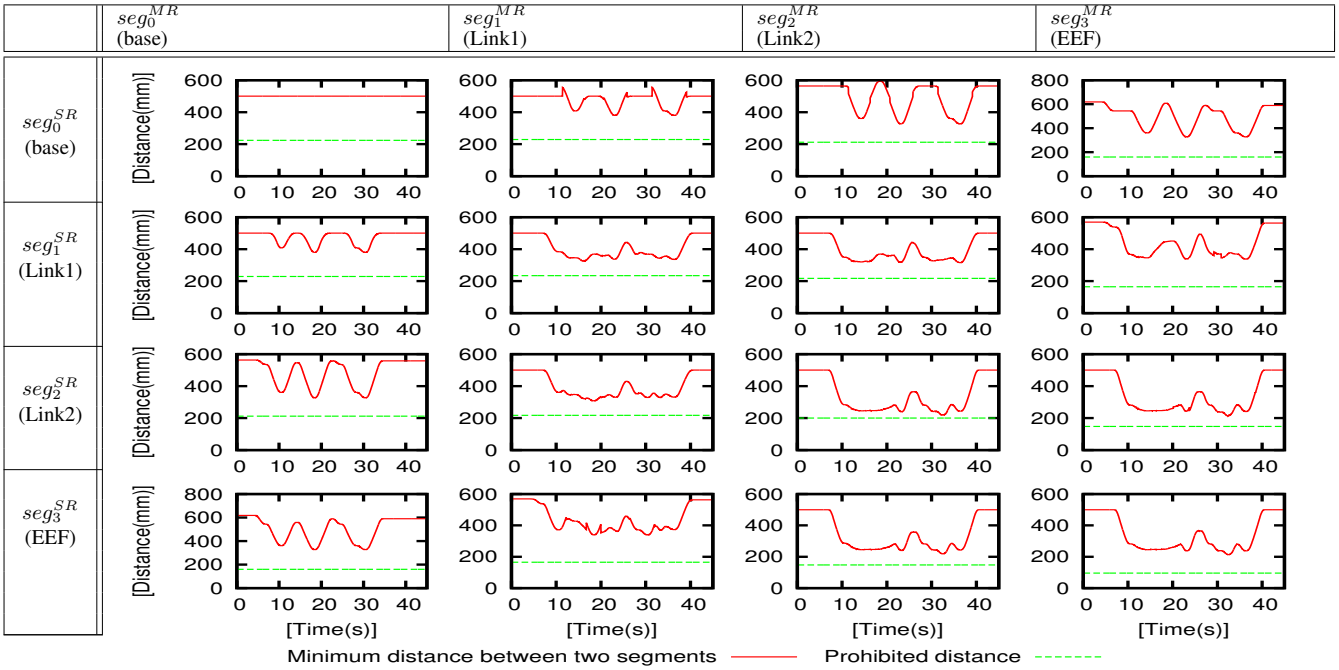


TABLE IV

TRACKING RESULTS OF MINIMUM DISTANCE BETWEEN THE LINKS OF BOTH ROBOTS AFTER APPLYING COLLISION AVOIDANCE SYSTEM



the maximum velocity and acceleration of each robot EEF are set to be $V1_{max} = V2_{max} = 100mm/s$ and $a1_{max} = a2_{max} = 100mm/s^2$ respectively. The initial posture of the robots' EEFs are $R1_{init}(450, 250, 300, 0, 0, 0)$ and $R2_{init}(450, -250, 300, 0, 0, 0)$.

In the first part of the experiment, the PTP commands are sent double times directly to the controller. Thus, the commands operation times from the start up of the system are illustrated in Fig. 13(a). It is worth mentioning that the period before executing the first command is the time required for initializing the system. The minimum distance between each pair of segments of the robots is tracked, and the results for all possibilities of pairs are acquired as shown in Table. III. The distance curve is illustrated in a solid red line, and the prohibited distance, which is the radii summation of spheres which model the tracked segments, is shown as a dashed green line. We can notice that several curves have passed through the prohibited distance, which means there

TABLE V

COMPARISON BETWEEN CURRENT AND PREVIOUS METHOD.

	Without collision avoidance system	**Previous method**	**New proposed method**
Method	None	Zone-blocking	collision map
Modelling	None	None	SSV
Time spent to execute commands	24.3[s]	40.7[s]	36.2[s]

are inevitable collisions.

The second part of the experiment has been done with including the collision & deadlock avoidance module. Thus, the commands operation times for this experiment are shown in Fig. 13(b). It can be noticed that there are two deadlocks have been avoided by adding two new commands. The results of the distance curve of each pair are illustrated in Table. IV. It is observable that all curves are navigating away from the prohibited distance. This means the system is able to avoid all potential collisions and deadlocks successfully.

B. Evaluation of Collision Avoidance System

By evaluating our new system, it is obvious that the current system is more advanced than [1]. It can detect and avoid collisions between all links of the whole bodies of the robots. In addition, it can avoid the deadlocks which cause blocking the motion of both robots.

A comparison with the work of Zhou *el al.* [19], a system which has the same problem formulation of ours, shows that the new system proves an advantage in time efficiency. The collision avoidance system of Zhou, which is zone-blocking, is simple and safe but the time wasted is large due to the inability of the robot to enter to the workspace if another robot is operating in it. The same experiment using the PTP commands which shown in Table II has been carried out using Zhou's method. The commands operation times of Zhou's methods are illustrated in Fig 13(c). Thus, the comparison results between both the proposed and Zhou's methods is shown in TABLE V.

VII. CONCLUSIONS AND FUTURE WORKS

This paper has proposed a novel concept for avoiding collisions and deadlocks between two n-link robot manipulators in an on-line system. The robots are controlled using PTP commands, and have no prior knowledge of commands to be sent. Therefore, the advanced collision map has been created for detecting potential collisions between the whole robot bodies. Furthermore, the map has been used skilfully to avoid the deadlocks which may happen if both robots become obstacles to each other. To decrease the computational cost of the collision detection, the robot links have been approximated geometrically using SSV with line primitives which is tight modelling for a near-tube robot shape. Collisions have been avoided using time scheduling of the execution time of PTP commands to avoid the collision area on advanced map. The simulations have demonstrated a successful avoidance of the collisions and deadlocks as well as an advantage in time efficiency comparing to previous method.

However, the algorithm is limited to check the collisions for two robots, Therefore, we are improving the current algorithm to be more effective to use for on-line collision avoidance of three or more robots.

REFERENCES

[1] A. Y. Afaghani and Y. Aiyama, "On-line collision avoidance between two robot manipulators using collision map and simple escaping method," in *Proc. IEEE/SICE Int. Symp. on System Integration*, 15-17 Dec 2013, pp. 105–110.

[2] R. Zurawski and S. Phang, "Path planning for robot arms operating in a common workspace," in *Proc. IEEE Int. Conf on Industrial Electronics, Control, Instrumentation, and Automation. Power Electronics and Motion Control*, 9-13 Nov 1992, pp. 618–623.

[3] L. Tsai-Yen and J.-C. Latombe, "On-line manipulation planning for two robot arms in a dynamic environment," in *Proc. IEEE Conf. on Robotics and Automation*, vol. 1, 21-27 May 1995, pp. 1048–1055.

[4] B. H. Lee and C. S. G. Lee, "Collision-free motion planning of two robots," *IEEE Trans. Systems, Man and Cybernetics*, vol. 17, no. 1, pp. 21–32, Jan 1987.

[5] C. Chang, M. J. Chung, and B. H. Lee, "Collision avoidance of two general robot manipulators by minimum delay time," *IEEE Trans. Systems, Man and Cybernetics*, vol. 24, no. 3, pp. 517–522, Mar 1994.

[6] P. A. O'Donnell and T. Lozano-Periz, "Deadlock-free and collision-free coordination of two robot manipulators," in *Proc. IEEE Int. Conf on Robotics and Automation*, vol. 1, 14-19 May 1989, pp. 484–489.

[7] Z. Bien and J. Lee, "A minimum-time trajectory planning method for two robots," *IEEE Trans. Robotics and Automation*, vol. 8, no. 3, pp. 414–418, June 1992.

[8] J. Lee, "A dynamic programming approach to near minimum-time trajectory planning for two robots," *IEEE Trans. Robotics and Automation*, vol. 11, no. 1, pp. 160–164, Feb 1995.

[9] S. W. Lee, Y. S. Nam, K. D. Lee, and B. H. Lee, "A safety arc based collision avoidance algorithm of a two-arm robot manipulator," in *Proc. 35th SICE Annual Conf. Int. Session Papers*, 1996, pp. 1167–1172.

[10] K.-S. Hwang, M.-Y. Ju, and Y.-J. Chen, "Speed alteration strategy for multijoint robots in co-working environment," *IEEE Trans. Industrial Electronics*, vol. 50, no. 2, pp. 385–393, Apr 2003.

[11] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 2, Mar 1985, pp. 500–505.

[12] S. Kalaycioglu, M. Tandirci, and D. Nesculescu, "Real-time collision avoidance of robot manipulators for unstructured environments," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 1, 2-6 May 1993, pp. 44–51.

[13] X. Cheng, "On-line collision-free path planning for service and assembly tasks by a two-arm robot," in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 2, 21-27 May 1995, pp. 1523–1528.

[14] R. Z. Lise Cellier, Pierre Dauchez and M. Uchiyama, "Collision avoidance for a two-arm robot by reflex actions: Simulations and experimentations," *J. of Intelligent and Robotic Systems*, vol. 2, no. 14, pp. 219–238, 1995.

[15] E. Freund and J. Rossman, "The basic ideas of a proven dynamic collision avoidance approach for multi-robot manipulator systems," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, vol. 2, 27-31 Oct 2003, pp. 1173–1177.

[16] P. Bosscher, D. Hendman, H. Corporation, and P. Bay, "Real-time collision avoidance algorithm for robotic manipulators," in *Proc. IEEE Int. Conf. on Technologies for Practical Robot Applications (TePRA 2009)*, 9-10 Nov 2009, pp. 113–122.

[17] R. G. Beaumont and R. M. Crowder, "Real-time collision avoidance in two-armed robotic systems," *J. of Computer-Aided Engineering*, vol. 8, no. 6, pp. 233–240, Dec 1991.

[18] A. Spencer, M. Pryor, C. Kapoor, and D. Tesar, "Collision avoidance techniques for tele-operated and autonomous manipulators in overlapping workspaces," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 19-23 May 2008, pp. 2910–2915.

[19] J. Zhou, K. Nagase, S. Kimura, and Y. Aiyama, "Collision avoidance of two manipulators using rt-middleware," in *IEEE/SICE Int. Symp. on System Integration.*, 20-22 Dec 2011, pp. 1031–1036.