

# On-line Path Planning for Assembly Operations by a Two-Arm Robot

X. Cheng

Institute for Real-Time Computer Systems and Robotics ( IPR )  
Prof. Dr.-Ing. U. Rembold, Prof. Dr.-Ing. R. Dillmann  
University of Karlsruhe, Kaiserstr.12, D-76128 Karlsruhe, Germany

## 1 Introduction

In the domain of autonomous assembly, intelligent control architectures have been proposed [1, 2, 3]. They consist of a hierarchy of functional levels, including planning, interpretation, execution and reaction, and incorporating sensor information. In such reactive systems, both strategic decisions and collision-free paths of manipulators have to be made / planned on-line, depending on the sensor information. The problem of on-line path planning is complicated by a two-arm robot with common work space since the robot arms become moving obstacles for each other. This paper focuses on the on-line path planning of a two-arm manipulator system for both independent tasks and two-arm cooperations, integrated in an on-line task-level planner.

Previous works related to path planning and coordination of multiple manipulators for independent tasks can be classified into local and global approaches. Local approaches [4, 5] fulfill the real-time requirements very well, but in the general case they can not guarantee that each manipulator will reach its goal because of the problem of local minima. Global approaches [6, 7, 8] are based on Configuration/ Time-Space. Due to their extremely high complexity, they are not suitable for on-line operation. Moreover, all of the global approaches assume that the motions of both manipulators are completely known for a period of time and the paths will be planned at once. But, in a hierarchical control architecture, goal positions of the manipulators are generated, in principle, asynchronously, and the on-line path planning has to generate a collision-free path, for example, for one manipulator while the other is moving. For archiving the goal states of the task-level commands, robot motions have to be planned automatically on-line and executed in parallel when possible.

In on-line operation of assembly tasks the time needed for robot motion planning is only acceptable when it is shorter or comparable with the time needed for motion execution. Therefore, a compromise must be made between the complexity and completeness of the algorithm for collision-free path planning. Completeness means hereby that the algorithm will always find a collision-free path if one exists.

## 2 System overview

At the Institute for Real-Time Computer Systems and Robotics of the University of Karlsruhe, a mobile two-arm

robot system is being developed. The robot KAMRO consists of three main components: the vehicle, the two PUMA260 manipulators in a hanging configuration, and the sensory system. The robot control system of KAMRO consists of two main parts: the on-line task-level planning system and the real-time robot control system which are located in different computer systems connected by a local area network[2].

The main function of the on-line task-level planning system is to interpret an implicit action plan of an assembly task and to generate sequences of explicit robot motion commands which are sent to and carried out by the real-time robot control system [9]. An action plan is a precedence graph, represented in a Condition/Event-Petri net, which describes an assembly task by specifying a set of pick-and-place operations, called Implicit Elementary Operations (IEOs), and their precedence relations. Considering the precedence relations, the work spaces of the manipulators, and the current locations of assembly parts on the work table, the IEOs are scheduled dynamically. For carrying out these IEOs, sequences of explicit robot motion commands, called Explicit Elementary Operations (EEOs), are generated and their execution by the real-time robot control system is monitored [10].

For cross-motions the on-line planning system specifies the paths as point sequences of the manipulators' Tool Center Point (TCP) in the Cartesian space. A linear interpolation of the Cartesian path points, the inverse transformation into the joint spaces, and the motion control is done by the real-time robot control system. For monitoring the execution of the planned sequences of EEOs, a concept for managing operation queues using waiting points is applied [10]. This concept synchronizes the EEOs by sequencing and temporally delaying their execution by the same system agent, and allows the parallel execution of EEOs by the different system agents when possible.

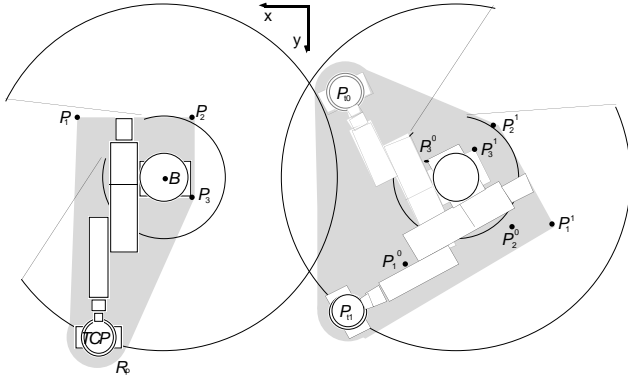
## 3 On-line path planning for independent tasks

The KAMROs two manipulators share a common work space, which enables two-arm cooperations. Because of the common work space, collisions can not only occur between manipulators and assembly parts, but also between the manipulators themselves during parallel execution of indepen

TCP motions approaching and departing parts are planned in the direction normal to the work table surface. In the proximity of the parts, force-guided fine motions are applied. Cross-motions of the TCPs in an x-y plane are only allowed in one layer, called the transportation layer, which is high enough above the work table so that the manipulator and the pay load can not collide with the assembly parts on the table. Therefore, to avoid collisions between the two manipulators, only the trajectories in the transportation layer need to be considered.

### 3.1 Swept regions of manipulators

The swept region of a manipulator for a cross-motion  $SR(M, PayLoad, Path)$  is defined as the polygon in the 2D geometric model swept out by the manipulator  $M$  and its pay load  $PayLoad$  as the TCP moves from  $P_s$  to  $P_g$  along the path  $Path = (P_s, \dots, P_g)$  (Fig. 1). The swept region is defined for the entire time period the manipulator needs to perform an EEO, and after its completion, the swept region will be updated to form the envelope of the manipulator and its pay load at the current position. The motion of a manipulator is guaranteed collision-free when its swept region is not touched by the opposite manipulator.



**Fig. 1** Work spaces and swept regions of manipulators in the 2D geometric model.

For determining the 2D envelope of a manipulator, we used three support points,  $P_1$ ,  $P_2$  and  $P_3$ , which are fixed in the base frame of each manipulator (Fig. 1). The pay load and the robot gripper are approximated by its circumference with the TCP as the center, represented by an N-vertex polygon. In the case of a single point path, the swept region  $SR(M, PayLoad, (P))$  is calculated using Graham's convex hull algorithm [11], which finds the convex hull polygon for a given set of points. Even though a manipulator does not change its TCP position during the execution of an EEO (e.g., grasp or detach), the swept region still has to be updated after the completion, because the circumference of the pay load may have been changed. When the path has more than one point, the polygon containing the path points and support points of the manipulator both at the start and goal position is determined first. The swept region is then calculated by expanding this polygon by the radius of the circumference of the pay load and the gripper. In the general case, the swept regions need not be convex polygons. They need not be simple, i.e. the edges of a polygon can intersect with each other. Because we use an intersection test

algorithm based on testing edge intersections of two polygons [11], these properties do not effect the result of the intersection test of the swept regions of the manipulators.

### 3.2 Planning collision-free paths

Suppose the manipulators are set up properly, i.e. their swept regions do not intersect at any point in time. The next EEO for execution in the operation queue of the manipulator  $M$  is a cross-motion from the current position  $P_s$  to the goal position  $P_g$ . The swept region of the opposite manipulator at this moment is  $SR_o$ . Based on the hypothesis of a "direct" path  $Path$  from  $P_s$  to  $P_g$  which consists of a single line segment, or a chain of line segments for going around the inner work space boundary, the following algorithm **Get\_CF\_Path** tries to plan a collision-free path for the manipulator  $M$  by verifying and modifying this "direct" path  $Path$ . The intersection test for two swept regions, " $\cap$ ", is done by using a line segments intersection test algorithm presented in [11].

```

Get_CF_Path( $M, P_s, P_g, Path, SR_o$ )
  Calculate  $SR(M, PayLoad, (P_g))$ ;
  if  $SR(M, PayLoad, (P_g)) \cap SR_o \neq \emptyset$ 
  then return( Collision( $P_g$ ) )
  else { Calculate  $SR(M, PayLoad, Path)$ ;
        if  $SR(M, PayLoad, Path) \cap SR_o \neq \emptyset$ 
        then Get_CF_Step( $M, PayLoad, ( ), Path$ );
        else return(  $Path$  ) }.

```

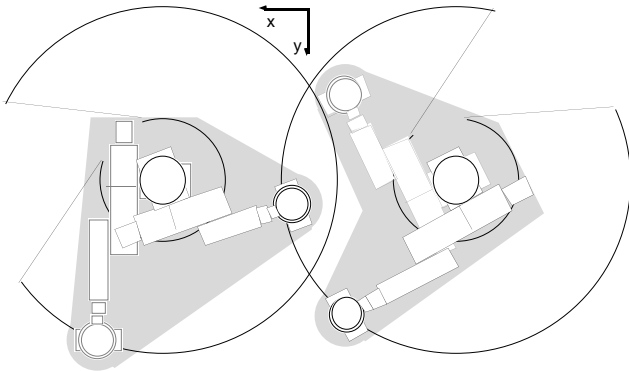
```

Get_CF_Step( $M, PayLoad, CF\_Path, (P_0, P_1, \dots, P_n)$ )
  if n=0 then {
    Path := append( CF_Path, ( $P_0$ ) );
    return( Path ); };
  if  $|P_0P_1| < Limit$  then return( Collision( $P_0$ ) );
  Calculate  $SR(M, PayLoad, (P_1))$ ;
  if  $SR(M, PayLoad, (P_1)) \cap SR_o \neq \emptyset$  then
  {    $P_1^c := \mathbf{Modify\_Path\_Point}(M, P_1)$ ;
      Calculate  $SR(M, PayLoad, (P_1^c))$ ;
      if  $SR(M, PayLoad, (P_1^c)) \cap SR_o \neq \emptyset$ 
      then return( Collision( $P_1$ ) )
      else  $P_1 := P_1^c$  };
  Calculate  $SR(M, PayLoad, (P_0, P_1))$ ;
  if  $SR(M, PayLoad, (P_0, P_1)) \cap SR_o \neq \emptyset$  then {
     $P_0^m := \mathbf{Get\_Inter\_Point}(M, P_0, P_1)$ ;
    Get_CF_Step( $M, CF\_Path,$ 
                ( $P_0, P_0^m, P_1, \dots, P_n$ ) ) }
  else { CF_Path := append( CF_Path, ( $P_0$ ) );
        Get_CF_Step( $M, PayLoad, CF\_Path,$ 
                    ( $P_1, \dots, P_n$ )); }.

```

The procedures **Modify\_Path\_Point**( $M, P$ ) and **Get\_Inter\_Point**( $M, P_1, P_2$ ) try to modify or generate new path point to avoid intersections between the swept regions (Fig. 2). The procedure **Get\_CF\_Step** stops when the length of the line segment  $P_0P_1$  is below some lower limit. This is the case when a sequence of short line segments converges

to a point where an intersection between the swept regions can not be avoided.



**Fig. 2** Avoiding collisions between two manipulators.

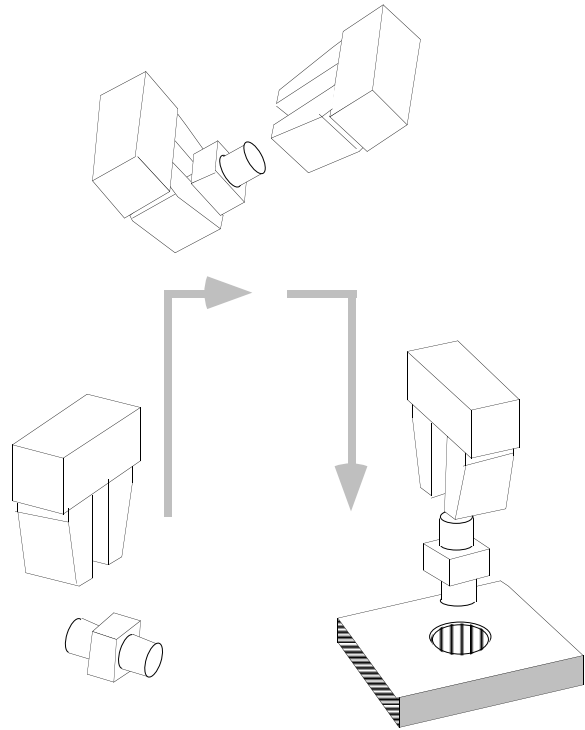
If a collision-free path is found, the corresponding swept region is reserved for the manipulator  $M$ . The cross-motion is started by sending the motion command to the real-time robot control system. Otherwise, the execution of the cross-motion is delayed by inserting a waiting point  $\text{wait\_for}(M_o)$  into the operation queue. Then the operation state of the opposite manipulator  $M_o$  is analyzed. If  $M_o$  is in the idle state, "direct" path is generated to bring  $M_o$  out of the common work space. Otherwise, if  $M_o$  is busy, the manipulator  $M$  goes into the idle state. The waiting point  $\text{wait\_for}(M_o)$  is removed from the operation queue as soon as the swept region of  $M_o$  has been changed, and the collision-free path planning is retried for the manipulator  $M$ . In this way, a dynamic scheduling strategy of "first coming, first executing" is realized.

Because of our simple model, algorithms of low computational complexity are applied for geometrical reasoning, so that, in the average case, the time of motion planning is comparable with that of the motion execution. Because the swept regions of the manipulators are defined for the entire time period of each cross-motion, collision-free paths can be planned asynchronously for the manipulators. In spite of delays, the cross-motions are guaranteed by the dynamic scheduling strategy to always reach their goal positions.

#### 4 On-line path planning for two-arm cooperations

KAMRO has to assemble a set of parts into a product according to a given assembly plan. The parts are allowed to be spread out randomly in an area on the work table which is visible via the overhead camera, and are reachable by at least one of the two manipulators. Therefore, the situation is possible that the assembly position of a part can only be reached by one manipulator, but the current position of the part lies in the work space of the other manipulator. Due to the geometry of the grippers and the kinematic restrictions of the manipulators, parts can sometimes only be picked up in a specific grasp configuration and must be assembled in a different one (Fig. 3). Two two-arm cooperations are defined to exchange and regrasp assembly parts. They are specified by the role of the manipulators, the part, and the different grasp configurations. The grasp configurations are

given as relative position and orientation of the TCP with respect to the part.

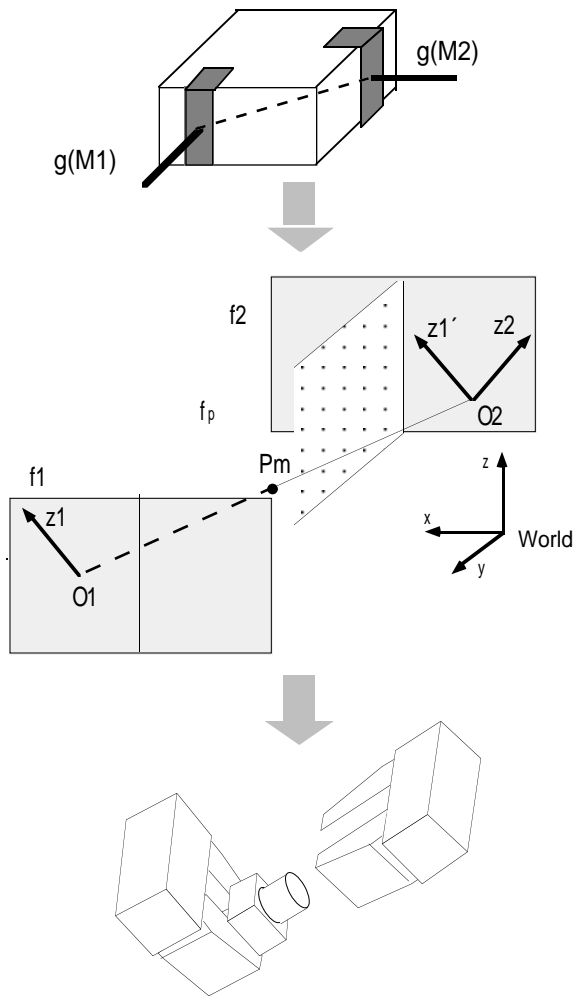


**Fig. 3** Exchanging / regrasping an assembly part using two cooperating manipulators.

On-line path planning for two-arm cooperations is done in two steps. First, the position and orientation of the part when it is being exchanged are determined, so that the two manipulators can reach the grasp configurations, from some safe positions out of the common work space, collision-free and without being restricted by their arm kinematics. Based on this result, paths of the manipulators for picking up the part and approaching their absolute exchange and regrasp positions are planned.

The position and orientation of the part when it is being grasped by the both manipulators are determined separately. The position in the common work space is obtained in the knowledge base, depending on the distance and angle between the two grasp configurations. These absolute positions stored in the knowledge base, associated with intervals of distance and angle between the grasp configurations are results of the off-line connectivity analysis that will be described in the next section. The orientation of the part is calculated analytically, so that the two grasp configurations are symmetric to the plane which separates the two manipulators at the middle of their bases, and perpendicular to the line connecting these bases, as shown in Fig. 4.

In addition to the path planning, the execution of the planned robot motions for two-arm cooperations must also be synchronized. For synchronization and collision avoidance, sequences of EEOs with waiting points are generated and executed.



**Fig. 4** Determination of orientation of the assembly part at the exchange position.

## 5 Off-line connectivity analysis

For on-line determination of the absolute exchange and regrasp configurations, information about all the reachable TCP configurations in the common work space is necessary. High computational complexity would be needed to deal with this information directly. Therefore, an off-line connectivity analysis is performed for providing a set of appropriate exchange positions of all the assembly parts used. For this purpose, the connectivity net of the manipulator TCP is defined as a network in a 6-dimensional Cartesian space, where the nodes represent reachable configurations of the TCP and the arcs stand for the connectivity between directly neighboring nodes. Because the manipulators at their exchange positions always have orientations in an  $x$ - $z$ -plane, as described in the last section, the dimension of the connectivity nets can be reduced to 4. The connectivity nets of the manipulators are constructed off-line by testing all nodes and arcs in the common work space, using a simulation of the motion control system.

For a given interval of distance and angle, the pairs of symmetrical TCP configurations are searched in the connectivity nets. Out of this set, the pairs, to which a linear approaching path from some safe positions out of the com-

mon work space exists respectively, are then selected. An interactive analysis tool with a graphical user interface was developed. Using this tool, a set of proper absolute positions are determined for representative intervals of distance and angle between symmetrical TCP configurations.

## 6 Conclusion

This paper addresses the on-line path planning of a two-arm manipulator system for both independent assembly tasks and two-arm cooperations. For parallel execution of independent tasks, collision-free paths are planned using a 2D geometric model and are based on a scheduling concept in consideration of the swept regions by the robot arms during their motions. Because of the simple model and the low computational complexity of the applied algorithms, the time for path planning is shorter, or comparable with that of motion execution. In spite of delays, the cross-motions are guaranteed to always reach their goal positions. The on-line path planning for two-arm cooperations for exchanging / regrasping assembly parts incorporates an off-line connectivity analysis, avoiding both collisions and kinematic restrictions.

## ACKNOWLEDGEMENTS

This research work was performed at the Institute for Real-Time Computer Systems and Robotics, Prof. Dr.-Ing. U. Rembold and Prof. Dr.-Ing. R. Dillmann, University of Karlsruhe, Germany. The work is funded by the "Sonderforschungsbereich Künstliche Intelligenz - Wissensbasierte Systeme" of the Deutsche Forschungsgemeinschaft.

## REFERENCES

- [1] Chochon, H.; Alami, R.: NNS, A knowledge-based on-line system for an assembly workcell, IEEE Int. Conf. on R. & A. (1986)
- [2] Hörmann, A.; Rembold, U.: Development of an advanced robot for autonomous assembly, IEEE Int. Conf. on R. & A. (1991)
- [3] Kelley, R. B.: Vertical integration for robot assembly cells, IEEE Int. Conf. on R. & A. (1986)
- [4] Freund, E.; Hoyer, H.: Real-time pathfinding in multi-robot system including obstacle avoidance, The Int. Journal of Robotics Research, Vol. 7, No. 1, 1988
- [5] Kathib, O.: Real-time obstacle avoidance for manipulators and mobile robots, The Int. Journal of Robotics Research, 5(1):90-98, Spring 1986
- [6] Erdmann, M.; Lozano-Perez, T.: On multiple moving obstacles, IEEE Int. Conf. on R. & A. (1986)
- [7] O'Donnell, P. A.; Lozano-Perez, T.: Deadlock-free and collision-free coordination of two robot manipulators, IEEE Int. Conf. on R. & A. (1989)
- [8] Roach, J. W.; Boaz, M. N.: Coordinating the motions of robot arms in a common workspace, IEEE Journal of R. & A., Vol. RA-3, No. 5, (1987)
- [9] Hörmann, A.: On-line planning of action sequences for a two-arm manipulator system, IEEE Int. Conf. on R. & A. (1992)
- [10] Cheng, X.; Kappey, D.; Schloen, J.: Elements of an advanced robot control system for assembly tasks, 5th Int. Conf. on Advanced Robotics (1991)
- [11] Preparata, F. P.; Shamos, M. I.: Computational geometry, Springer Verlag, New York, 1985