Best Placement of Telescoping Robots in Environments with Obstacles

Krasimir Kolarov and Bernard Roth
Design Division, Mechanical Engineering Dept.
Stanford, CA 94305
E-mail: Krasi@Flamingo.Stanford.Edu

1. Introduction and definition of the problem

In robotics we are often concerned with the ability of certain robots to operate in an environment with obstacles. The basic problem we are discussing in this paper is the following: if we are given an environment with obstacles in it, we want to find the best placement and the most appropriate structure for a robot that operates in this environment, so that the robot can reach every point in the environment.

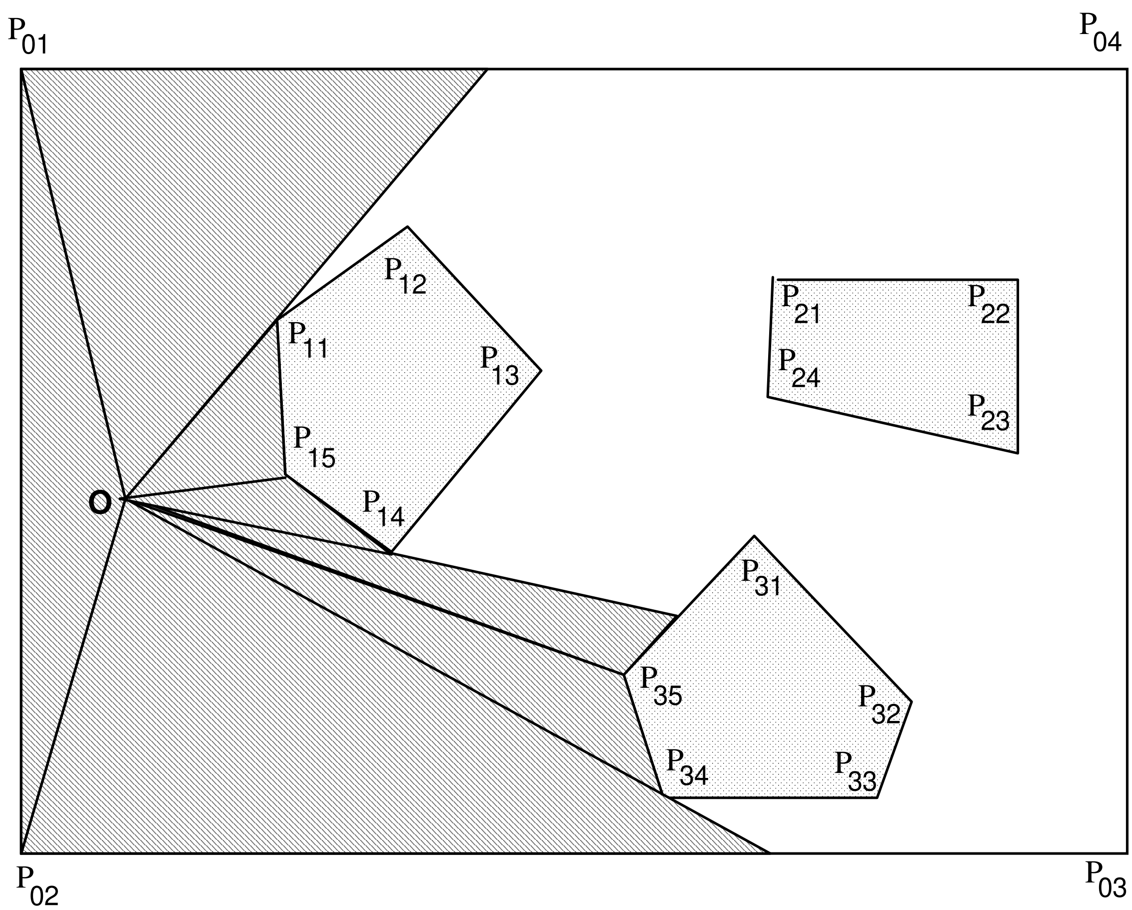

Figure 1. Visibility of a single link with telescoping structure

The types of robots that we are designing are fixed in the environment, and all their links have telescoping structure (see [1]). In those robots every link is connected to the next one by a revolute joint that allows it to rotate about its axis, while the links themselves stretch in and out like a telescope. Such links are well suited for planar types of problems. For example, as illustrated in Figure 1, with one telescoping link, fixed at point *O*, we can reach the whole shaded region on the figure. Due to the revolute joint we can reach concentric parts of circles, depending on the length of the link. Alternatively the prismatic joint allows for reaching radial segments, starting from point *O*, corresponding to the different angles of the link with the horizontal axis.

With this general structure of our robot, we are interested in two basic questions:

A. Where should we place the fixed base of the robot so that all points in the environment can be reached in an optimal manner?

B. Once we have chosen the place for the base, what is the minimal number of telescoping links that our robot should have so that it can reach all points in the environment?

In [1] we have developed an algorithm that solves problem B. This algorithm assumes that we are given a point where the base of the robot is fixed, and from this point we find iteratively the sets of points that can be reached with one link, two links, three links, etc. These sets are called "visibility regions" and the algorithm builds them until all the points in the free space *E* (those within the outside boundary, not belonging to the obstacles) are covered. In [3] that algorithm is shown to be optimal with a complexity *O(m.n.log(n))*, where *m* is the number of obstacles and *n* is the number of vertices (of the obstacles and the outside environment).

Different base points for our robot will have different "visibility" (minimum number of links needed to reach all other points in the free space). In problem A we want to find the set of points in the free space, that have minimal visibility overall. This set is called "link center" and the corresponding visibility of the points in the set is called "link radius".

The notions of visibility, link center and link radius are defined and used within the area of computational geometry mainly for simple polygonal environments with no obstacles (see [4], [5] and[6]).

In this paper we present an algorithm that builds this link center and some approximations to it. We introduce the algorithm for two-dimensional convex obstacles and then extend our discussion to non-convex, curvilinear and general 2D obstacles. We also consider extensions of our problem to higher dimensional spaces and to the problem of simultaneous design of the robot and the environment, i.e. when the obstacles are not fixed in the environment and we want to find their best placement within it.

2. Environment with convex polygonal obstacles

In [1] we have shown that if we can reach all points on the edges of the obstacles and the outside environment from a given fixed point with k number of links, than we can reach all points in the free space from this fixed point with k links. This criterion is also the base of the algorithm for finding the link center of the environment. In particular we build the visbility regions for all edges in the problem and then intersect the regions with the same visibility in increasing order until we obtain a non-empty intersection. This intersection is the link center and the corresponding visibility is the link radius.

We will define first the notion of the "k-th visibility region of an edge". In this region we include all points in the free space that can see the entire edge with maximum k links. For example in Figure 2 the first visibility region of the edge $e = P_{12}P_{13}$ is shaded vertically. This set is obtained as the intersection of the first visibility regions of point P_{12} and point P_{13}.

In general, as explained in [1], we construct the visibility regions for different levels by drawing tangents from characteristic ("generating") points of the previous level. Such points in Figure 2 are the points Q, R, S, T and U. Thus to find the second visibility region of edge e we need to find all the points that can see at least one of the above mentioned points with one link, i.e. we build the union of the first visibility regions for those points. If we do that the only points in the free space that are not included in this second visbility region, are those belonging to the triangles D and F.

However we can show that the point in triangle D can actually see the entire edge e with two links although they do not see any of the generating points Q, R, S, T and U with one link. The points in D can see points X and Y with one link, and X and Y together see entire edge e with one link. That is why the visibility of the points in D is two.

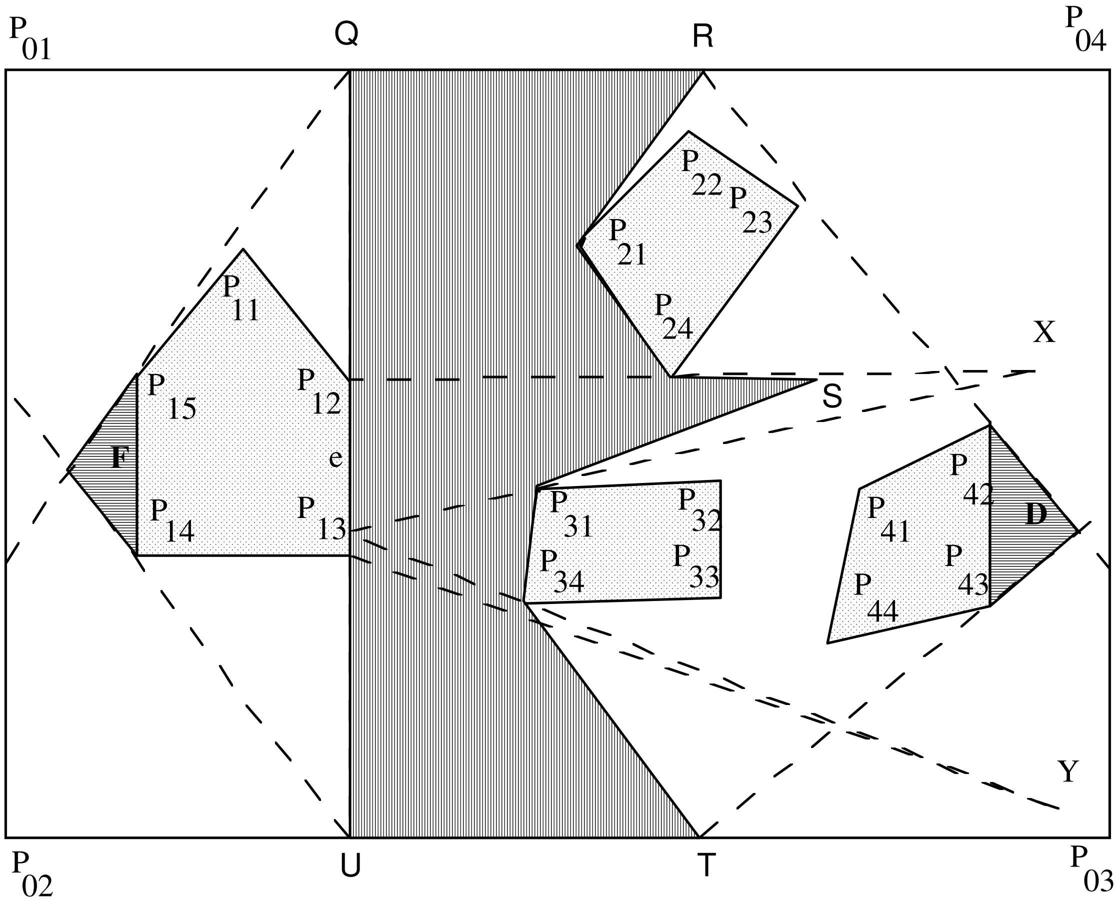

Figure 2 Visibility of an edge.

To account for those situations, we have introduced in [3] the geometric concept of an 'X-form' based on an edge. The terminology for the X-form is illustrated in Figure 3. The part of the edge that we want to see will be called the "base" of the X-form and the opposite side of the form will be called the "top". The tangents that outline the X-form are called "generating lines" of the form, and their intersection point V is the "vertex" of the X-form. The part inside the X-form between the vertex, the generating lines and the top is denoted as the "upper part", and the corresponding part for the base is called the "lower part" of the form. Any line l that intersects the lower part, passes through the vertex, or intersects the upper part but not the top of the X-form, is said to "cross" the form. Let U be a point in the free space E for which there exists a line l that crosses the X-form. Any such point U will be two-visible of segment e, i.e. it can see the entire segment with maximum two links. It is clear that all points in the lower part, including the vertex of the X-form, are one-visible of the base. Analogously all points in the upper part, including the top, are two-visible of the base segment of the form.

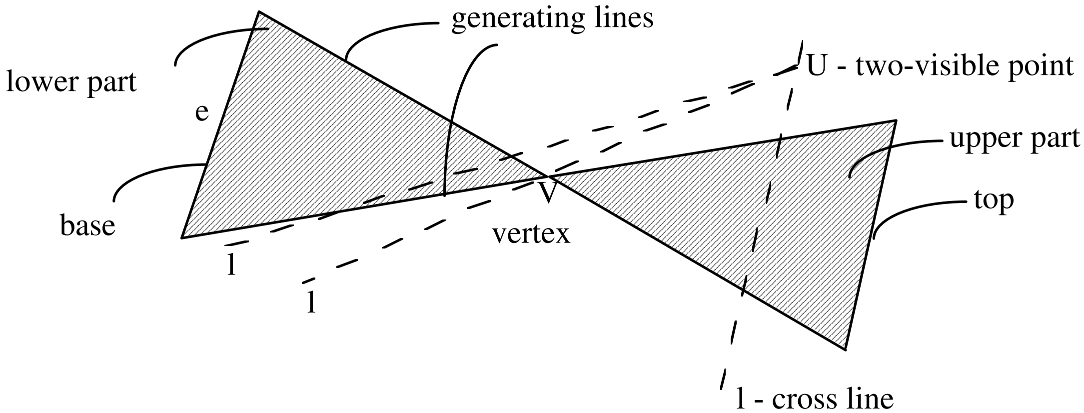

Figure 3. X-form of a segment

We can use the X-forms, by initially drawing all mutual tangents between the obstacles. For each edge we generate the corresponding X-forms by considering the tangents from the vertices of the edge to the other obstacles, and the mutual tangents that intersect the edge. All X-forms for a given edge are based on parts of the edge (or on the entire edge). A set of "complementary" X-forms have bases, whose union is equal to the entire edge.

The overall link center for the environment is built using the following algorithm:

begin
k:= 0;
repeat
    k:= k+1; {*k* is the current visibility}
    S_k*:=E;* {*E* is the whole free space}
    for *i=1,...,m* ***do*** {*m* - number of obstacles}
      for *j=1,...,m[i]* ***do*** {*m[i]* -number of vertices of ith obstacle}
        begin
        Build visibility region $V_k(e_{ij})$;
    {e_{ij} denotes the edge $P_{ij}P_{ij+1}$ }
        $S_k = S_k \cap V_k(e_{ij})$;
        ***end*;**
until S_k is non-empty ;
***end*;** {S_k is the exact link center}

In the procedure for building the visibility region $V_k(e_{ij})$ we first build the part of the region that corresponds to the generating point at level *k* and check if all points in the free space are covered. If that is the case we are done. Otherwise there are still edges in the environment that are not completely seen. If the lines, corresponding to any of those edges cross complementary X-forms of edge e_{ij}, then the appropriate edge is added to the visibility region $V_k(e_{ij})$.

The link center for our example is illustrated in Figure 4. There, the link center is the unshaded part of the free space. The complexity of our algorithm for the exact link center is *n* times the complexity of the algorithm for visibility from a point, described in [1].

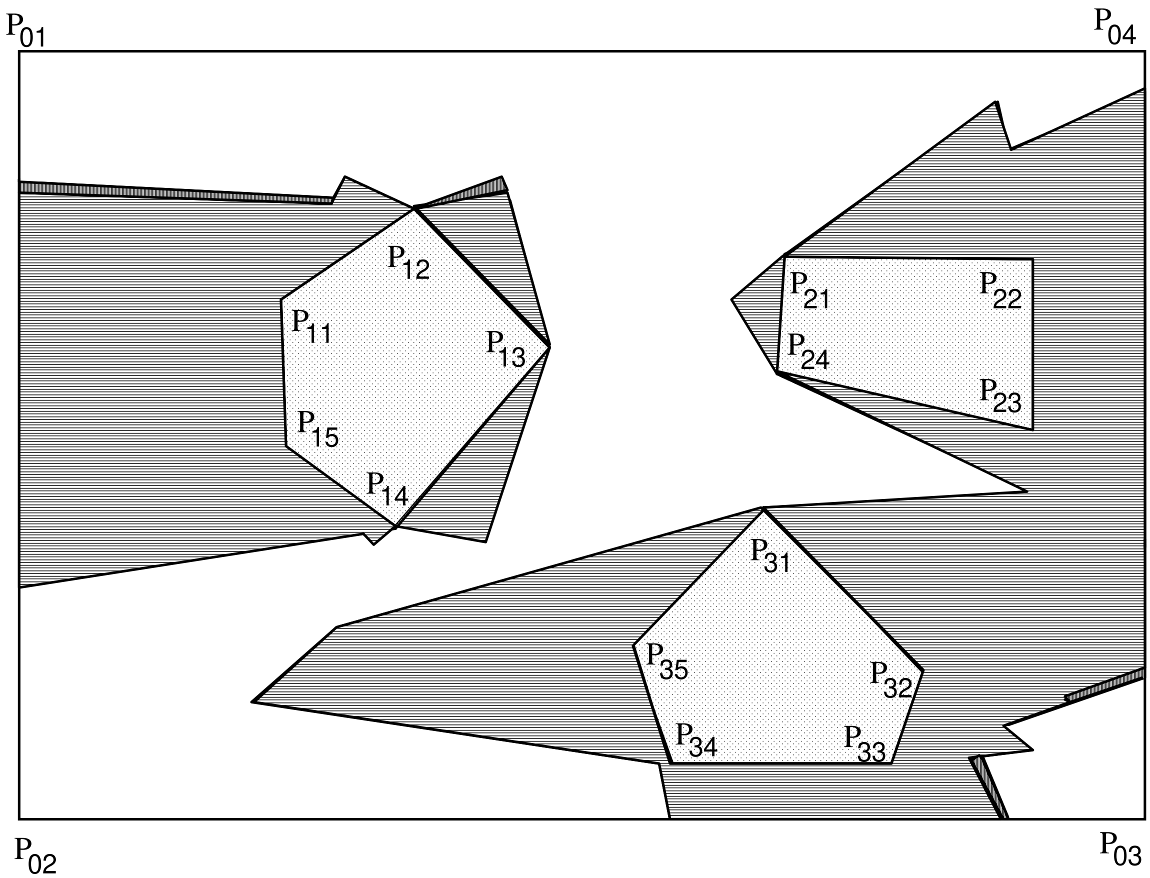

Figure 4. Exact and approximate link centers.

Because the implementation of the algorithm above with X-forms is quite complicated, we will also demonstrate an algorithm for finding an approximate link center (ALC) for the environment. ALC is a set of points for which the maximal "link distance" to any other point in the free space is at most one more than the minimal one for the environment (link distance between two points is the minimal number of links that are need to reach from one point to the other).

The ALC that we are interested in, is formed by all points that can see all vertices (or all vertices and midpoints) of the edges in the environment with minimum number of links. This ALC is built similarly to the exact link center, where instead of building the visibility regions of an edge, we build the visibility regions of the vertices (and eventually the midpoints) of the edge. This way we do not have to build X-forms and make complicated geometric constructions. The algorithm for ALC still has complexity of n times the complexity of the algorithm for visibility from a point, because for every vertex we can find an example where the contribution of this vertex to the ALC is significant.

In Figure 4 the ALC corresponding to the vertices and the midpoints of the edges is depicted as the unshaded area plus the four small vertically shaded areas. As we can see this is a good approximation to the exact link center.

3. Non-convex, curvilinear and general 2D obstacles

In this part we will outline how the algorithms described above, must be adapted to cover the case of general 2D obstacles.

If we have non-convex vertices in the polygons, we can build the visibility regions corresponding to them, using our algorithm for visibility from a point. Then an ALC for non-convex polygonal obstacles is built exactly the same way as the ALC for convex polygons.

To build the exact link center for a problem with non-convex obstacles we need to deal with visibility of reflex edges in the environment. However as we can see in Figure 5 the reflex pockets in the obstacles actually make our task easier. It is easy to see geometrically that everything that can be seen from the reflex edges $P_{13} P_{14}$, $P_{14}P_{15}$ and $P_{15}P_{16}$, can bee seen from the vertices of the reflex pocket P_{13} and P_{16}. Thus the non-convex edges of the obstacles actually do not contribute to the exact link center.

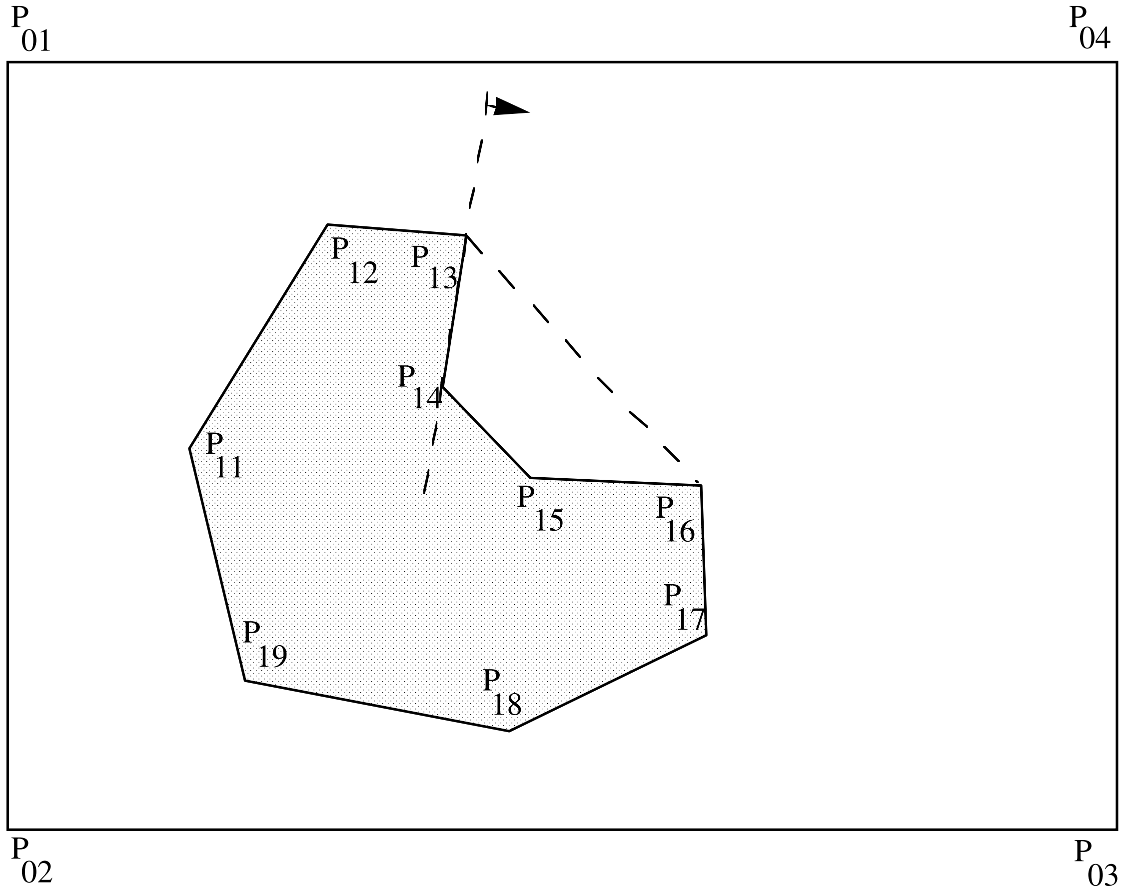

Figure 5. Visibility of reflex pockets from convex vertices

Using the argument above, we can show that the exact link center for non-convex obstacles is the intersection of the exact link center for the convex hulls of the obstacles, and the ALC for the vertices of the non-convex obstacles. The complexity of the algorithm for an ALC and the exact link center remain the same as before. The exact link center for non-convex obstacles is illustrated in Figure 6, where it is depicted again as the unshaded area on the figure.

If we allow circular (or curvilinear in genearl) edges in the convex polygons, the resulting obstacles are called generalized polygons. The theory for generalized polygons is analogous to the treatment of convex polygons. The visibility region from a point in an environment with obstacles represented as generalized polygons, is formed by drawing the tangents from the point to the obstacles and using again the mutual tangents between the obstacles. The only difference is that now we also have tangents from the points to arcs of circle. While the equation of the tangent in the case of convex polygons follows simply from the fact that the line passes through two known points, for generalized polygons this equation is a little more difficult to obtain. The corresponding equations, for this case, are derived in [3].

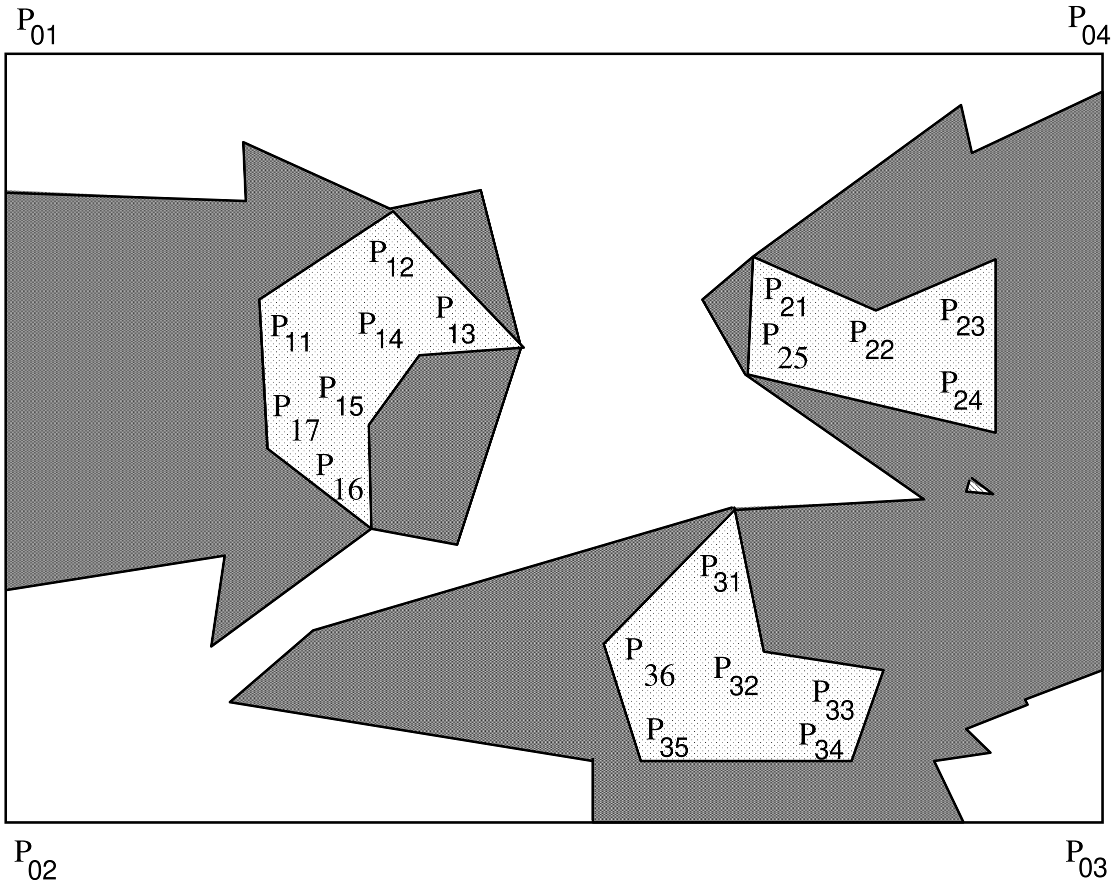

Figure 6. Exact link center for non-convex polygonal obstacles.

Using those formulas we can easily implement an algorithm for finding an ALC for generalized polygons. For the exact link center, however, we need additional constructions. The problem here is that, as opposed to the linear case, there might be no points in the free space that can see the whole curvilinear edge with one link only, i.e. the first visibility region of an arc can be empty.

This situation is illustrated in Figure 7. There edge e_{31} cannot be seen entirely from any point in the free space. To deal with this situation, we subdivide this edge in two half-edges and build the first visibility regions for each of them. In the figure those are represented with the horizontally shaded areas H'_{31} and H''_{31} resp. Then the second visibility region for the edge e_{31} consists of all points in the free space that can see at least one point form each of the visibility regions for the half-edges. In addition with second and higher level visibility regions, we need to consider X-form similar to those for convex polygons. The difference is that, now the base and/or the hat of the X-form can be circular, rather than straigth edges. In Figure 7 the third (and last) visbility region is the completely shaded region on the figure. The visibility of edge e_{31} is thus three.

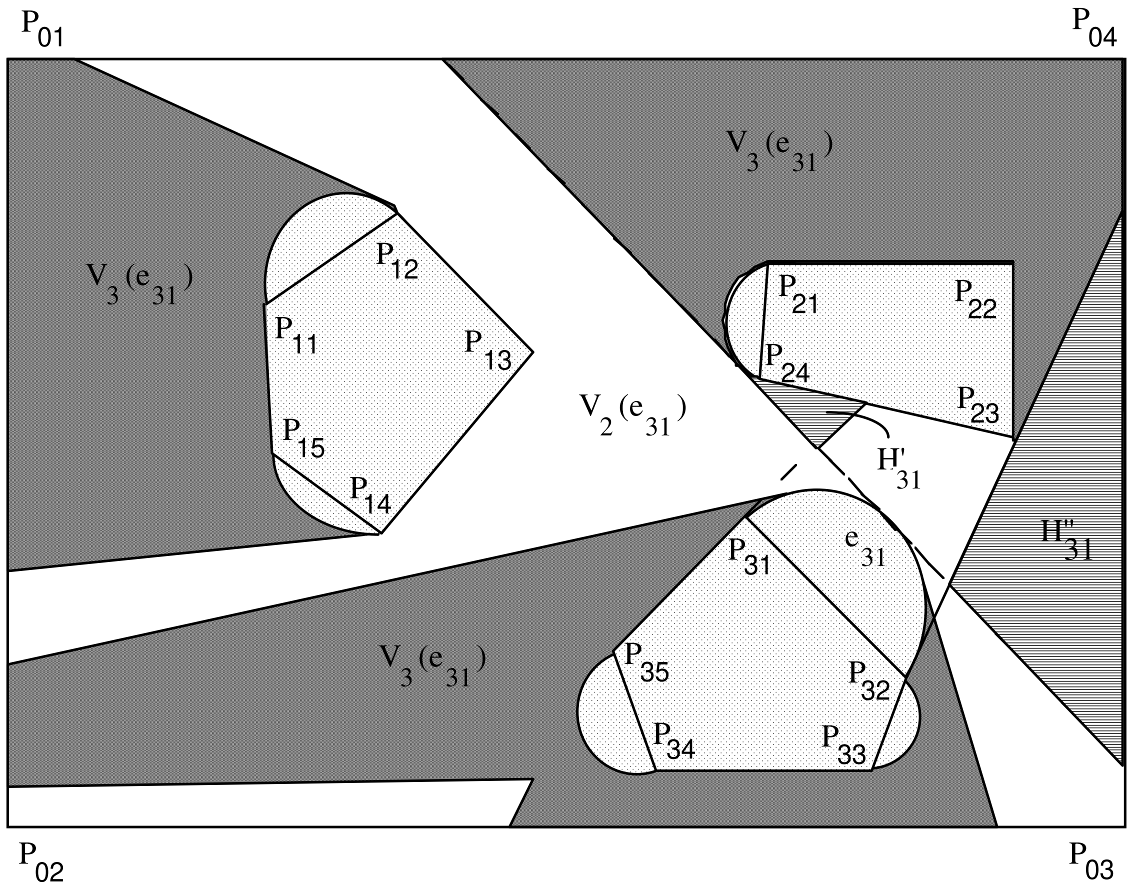

Figure 7. Visibility of the halves of an arc of circle

We illustrate the exact link center for generalized polygonal obstacles in Figure 8. There the obstacles are the same as the ones in Figure 7 (i.e. vertices P_{22}, P_{14} and P_{15} are exlcuded), and the link center is the unshaded area plus the horizontally shaded triangle. As we can see curvilinear edges considerably reduce the link center for the environment compared to straight ones.

If we combine our discussion of concavities and curvilinear edges, we can extend the algorithms for exact and approximate link centers to general two-dimensional obstacles. This case is illustrated in Figure 8, where now all vertices are included in the obstacles. The resulting exact link center is the entirely unshaded area on the figure. As we have noticed before, simple concavities do not affect the link center significantly.

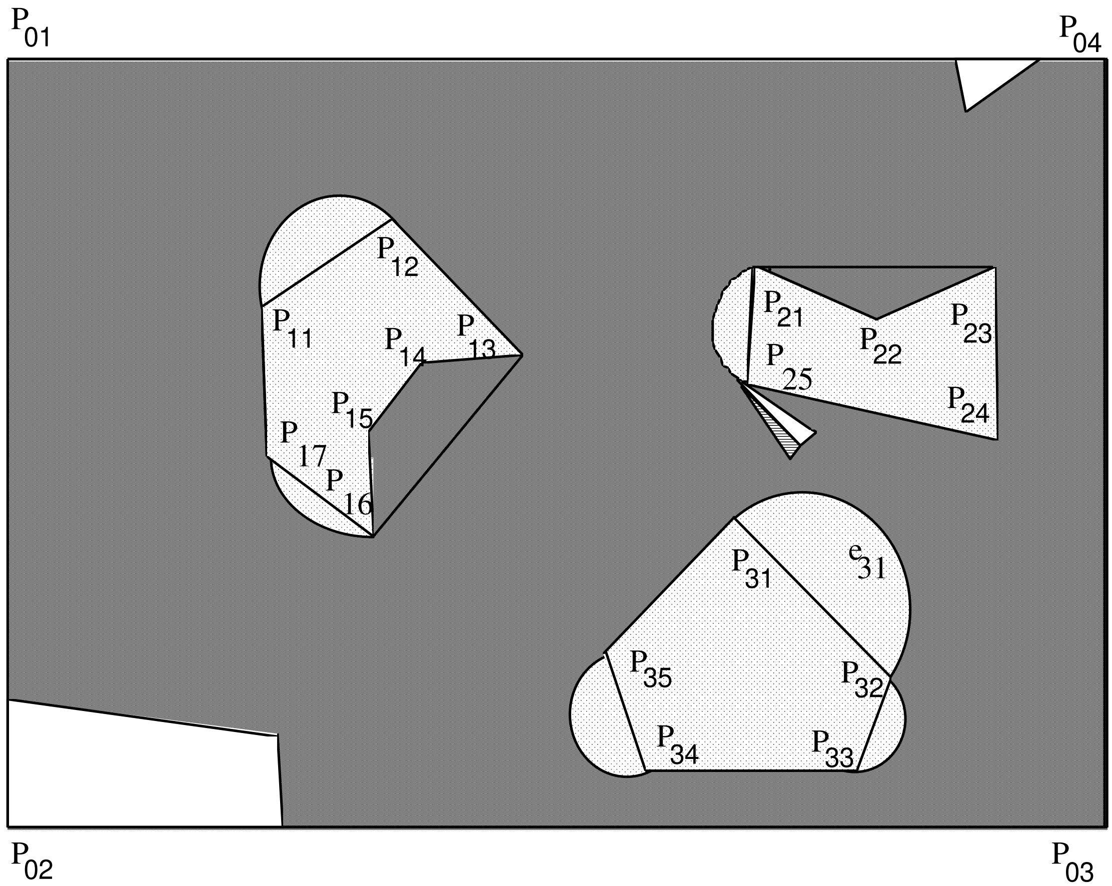

Figure 8. Exact link center for general planar obstacles.

4. Extensions and conclusions

4.1 Three-dimensional environment.

In three-dimensional environment we can extend our problem by considering links that have three degrees of freedom: two rotational freedoms on the end toward the base, followed by a translational telescopic joint. In this case we can model the obstacles as polyhedra and build the visibility regions in 3D. Those regions can be constructed using a tangential approach that includes tangent planes in addition to the tangent lines. More details on the general 3D case are outlined in [3].

We can easily extend our algorithms to $2^{1/2}$D, where the obstacles are considered somewhat uniform in vertical dimension. In the example of Figure 9 we can model the environment with a set of horizontal slices that describe the major geometrical changes in the objects. For example the coffee table in the figure is described with horizontal slices at heights l_1, l_2 and l_3 . For each slice we build the link center (Figure 10 for height l_3) and then project and intersect those link centers on the base plane. The result is an ALC, where we can place the base of a robot, whose first link is a prismatic one, that translates up and down in the environment to carry the rest of the structure to the appropriate height. For successful implementation of this algorithm, we need to be able to choose a sufficient set of horizontal slices that describes completely the obstacles in the environment.

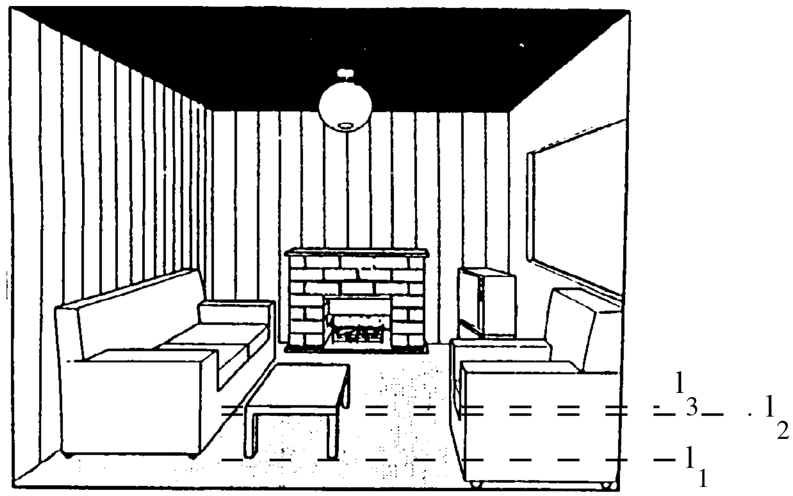

Figure 9. Horizontal slices in a 3D environment.

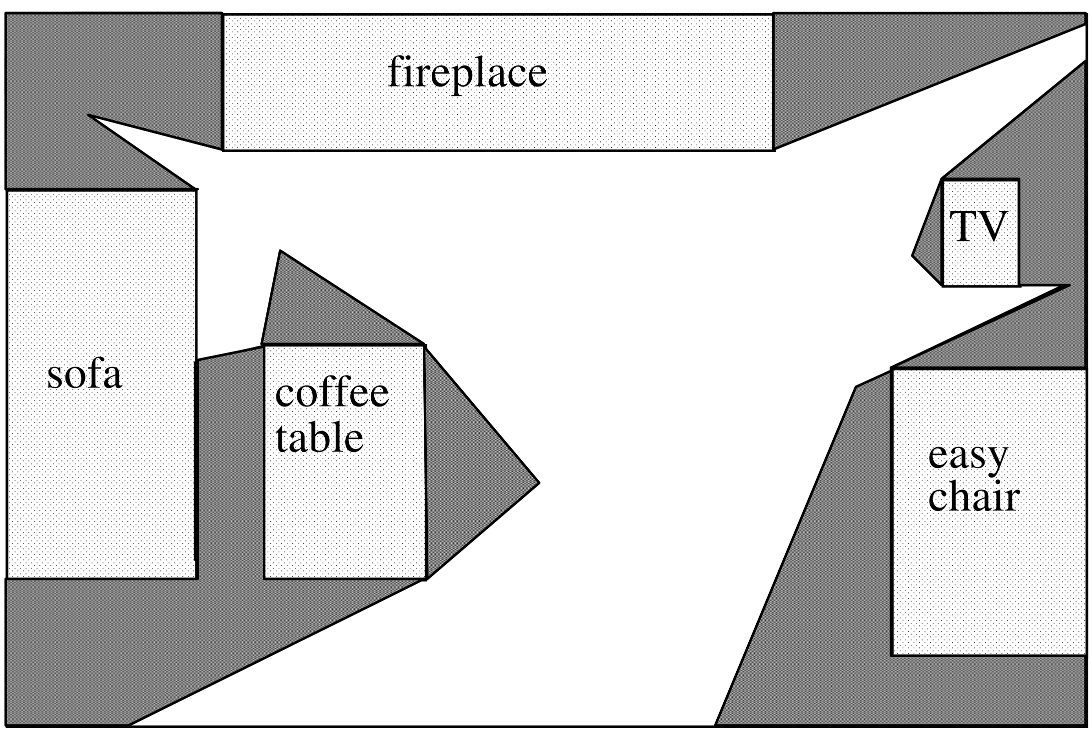

Figure 10. Link center for the slice at height *l_3*.

If there are holes or concavities in the objects, we can use slices in all directions, i.e. front - rear, top - bottom and left side - rigth side slices, to describe better the shape of the obstacles.

4.2 Simultaneous design of robot and environment.

If we have an environment and a set of obstacles that are not yet placed in the environment, we should try to place the obstacles in such a way, that the robot that will work in this environment can have an optimal structure to reach everywhere within the environment. In other words we can design both the environment and the robot simultaneously. One simple illustration of this approach is shown in Figure 11. There the obstacles are approximated with their minimal bounding rectangles (using for example rotating calipers) and are placed in a rectangular environment. As a result a robot with only <u>three prismatic</u> links can cover the whole free space.

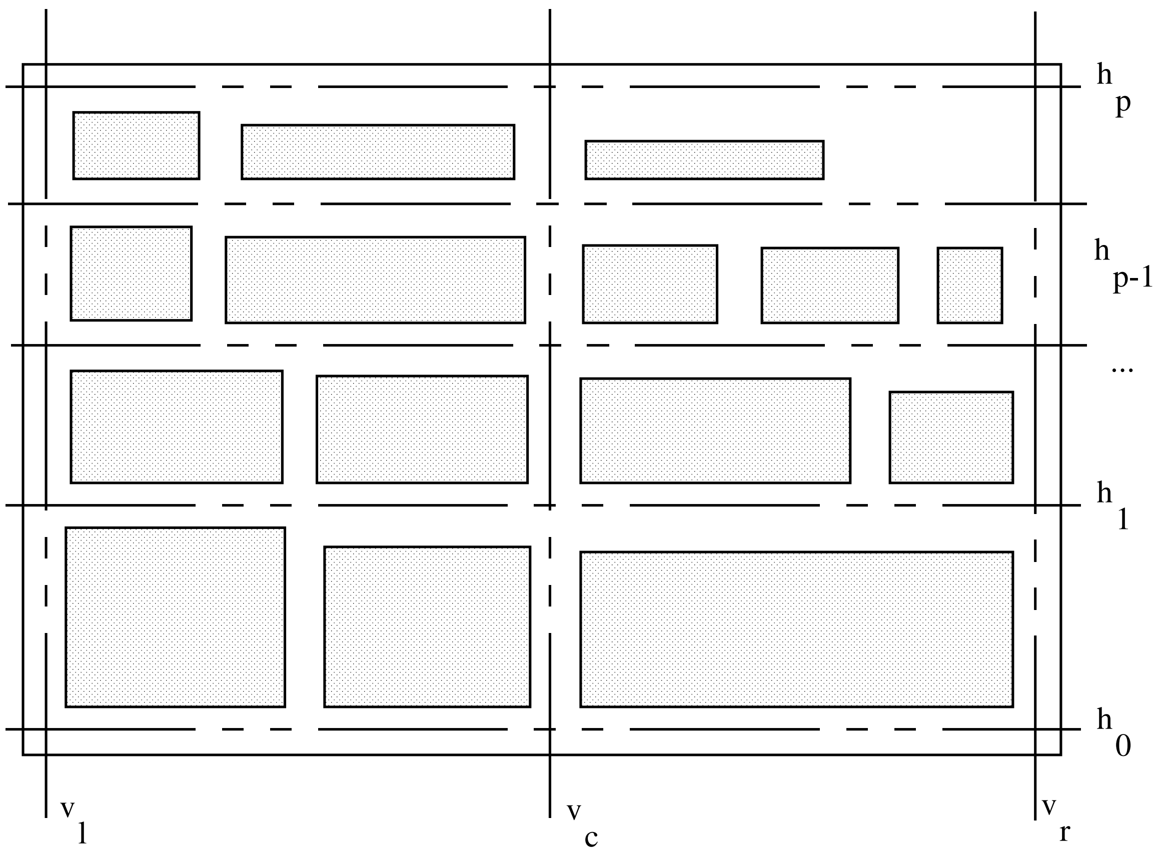

Figure 11. Simulataneous design of robot and environment.

More details on best placement of obstacles and robot in the environment can be found in [3].

4.3 Final remarks and conclusions.

The results we have presented so far have algorithmic character. In [2] we have derived some theoretical estimates of the minimum number of links needed for a given problem, that depend only on the number of obstacles *m* and the number of vertices of the obstacles and the outside environment *n* .

Other extensions of the problem, considered in [3], include moving obstacles and robot, multiple robots, reconfigurable structures and higher dimensional analysis. Interesting application areas include construction, factory automation and piping.

References:

[1] K.Kolarov and B.Roth, "On the Number of Links and Placement of Telescoping Manipulators in an Environment with Obstacles," *Proc. 4th International Conference on Advanced Robotics*, 1991, pp. 988-994.
[2] K.Kolarov and B.Roth, "Best Estimates for the Construction of Robots in Environments with Obstacles," *Proc. of the IEEE International Conference on Robotics and Automation*, France, 1992, pp.377-382.
[3] K.Kolarov, *Geometric Design of Robots for Environments with Obstacles*, PhD Thesis, Department of Mechanical Engineering, Stanford University, 1992.
[4] W.Lenhart, R.Pollack, J.Sack, R.Seidel, M.Sharir ,S.Suri, G.Toussiant, S.Whitesides and C.Yap, "Computing the Link Center of a Simple Polygon," *Proc. 3rd Annual ACM Symposium on Computational geometry*, 1987, pp.1-10.
[5] S. Suri, "On Some Link Distance Problems in a Simple Polygon," *IEEE Transactions on Robotics and Automation*, Vol.6, No.1, February 1990, pp.108-113.
[6] J.Mitchell, G.Rote and G.Woeginger, "Minimal Link Paths Among Obstacles in the Plane," *Proc. 6th Annual ACM Symposium on Computational Geometry*, 1990, pp. 63-72.