Localization with Few Distance Measurements

Dan Halperin Steven M. LaValle Barak Ugav

February 26, 2022

1 Introduction

A depth sensor is placed inside a known polygonal workspace W with n vertices. Consider the depth mapping $d = h(x, y, \theta)$ in which $(x, y) \in W$ is the position of the sensor and $\theta \in S^1$ (namely $[0, 2\pi)$) is the direction of the ray emanating from the sensor and along which the distance to the workspace boundary is measured. Thus the configuration space (C-space for short), namely the parametric space of sensor placements, is three-dimensional.

Problem statement

We wish to devise a data structure such that, after preprocessing the workspace, will efficiently answer the following queries: Given a distance measurement d as query, report all the possible sensor configurations that could yield such a measurement. In other words, we aim to compute h's preimage $h^{-1}(d) = \{(x, y, \theta) \in W \times S^1 \mid d = h(x, y, \theta)\}$.

Motivation

Such scenario can be applicable in real world cases, for example: a rover lands on Mars, a map of whose terrain is available to it. It looks about its position, and then infers its exact position on the Martian surface. Another application comes from robots that follow a planned path through a scene: the control systems that guide such a robot along the planned path gradually accumulate errors due to mechanical drift. Thus it is desirable to use localization from time to time to verify the actual position of the robot in the map, and apply corrections as necessary to return it to the planned path.

Our contribution

TBC

2 Data Structure for Single Distance Measurement

We describe a data structure that for a given sensor reading d, efficiently determines all the sensor placements that would result in this reading. We present the construction of the data structure in two stages. In Section 2.1 we describe a decomposition of the three-dimensional C-space of the sensor into cells of constant descriptive complexity each, such that within each cell C, the sub-region $C_d = \{(x, y, \theta) \in C \mid d = h(x, y, \theta)\}$ also has constant descriptive complexity . We then show, in Section 2.2, how to maintain the cells of the decomposition in a search structure, such that given a query d we can easily retrieve all the cells C for which $C_d \neq \emptyset$.

2.1 Decomposing the C-Space of the Sensor

Suppose for the purpose of exposition that we know that the sensor measures its distance in the upward vertical direction, namely $\theta = \pi/2$. We apply trapezoidal (vertical) decomposition [1, Chapter 6] to the workspace. Given a measurement d, we can easily compute for each trapezoid[1] in the decomposition if there are any points in it that are at vertical distance d from the top edge of the trapezoid. These points lie along a segment parallel to the top edge, and the desired answer is the union all such segments over all trapezoids. A trapezoid is identified by its ceiling and floor (namely top and bottom edges, respectively), and the left and right vertices, which define the left and right artificial vertical edges. See Figure 1.

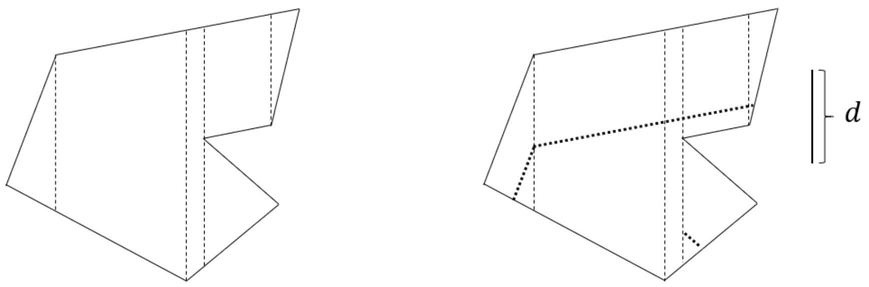


Figure 1: Example of vertical decomposition (a) and the line segments (bold, dotted) along which the sensor may be when it reads the distance d in the upward vertical direction (b).

To support different orientations of measurement, we need to apply a similar procedure to the scene for every orientation. For each orientation we decompose the workspace such that the imaginary walls are drawn in the respective orientation, and return the union of all the results. As we let the orientation θ of the measuring ray vary a little, and accordingly decompose the scene in the direction of the measurement, we notice that the trapezoids remain similar (in a sense to be made precise shortly) unless they undergo some combinatorial change. As the direction of measurement changes, the trapezoid changes continuously, and we refer to the union of these variations as a three-dimensional cell, or *cell* for short. Since throughout the change of orientation the ceiling and floor edges, as well as the vertices determining the left and right boundaries of the trapezoid remain the same we use the following notation: A cell C is identified by its ceiling and floor (namely top and bottom edges, respectively), e_t^C, e_b^C, and the left and right vertices, v_l^C, v_r^C, which define the left and right artificial edges (parallel to the direction of measurement). The two limiting vertices, v_l^C, v_r^C, may be endpoints of e_t^C, e_b^C, or some other vertices. See Figure 2 for an illustration.

Now, in addition to describing a cell by its floor and ceiling, and the two vertices that define the two artificial side edges, we also add the first and last orientations where the

[1]Some of the two-dimensional cells in the decomposition may be triangles; for brevity we will also refer to them in this context as trapezoids.

trapezoid exists (in the combinatorial sense, namely having the same edges and vertices defining it). These cells constitute a decomposition of the C-space of the sensor. With this description, there are only $O(n^2)$ unique cells, and we can compute each cell's exact boundary as a function of the orientation θ of the sensor.

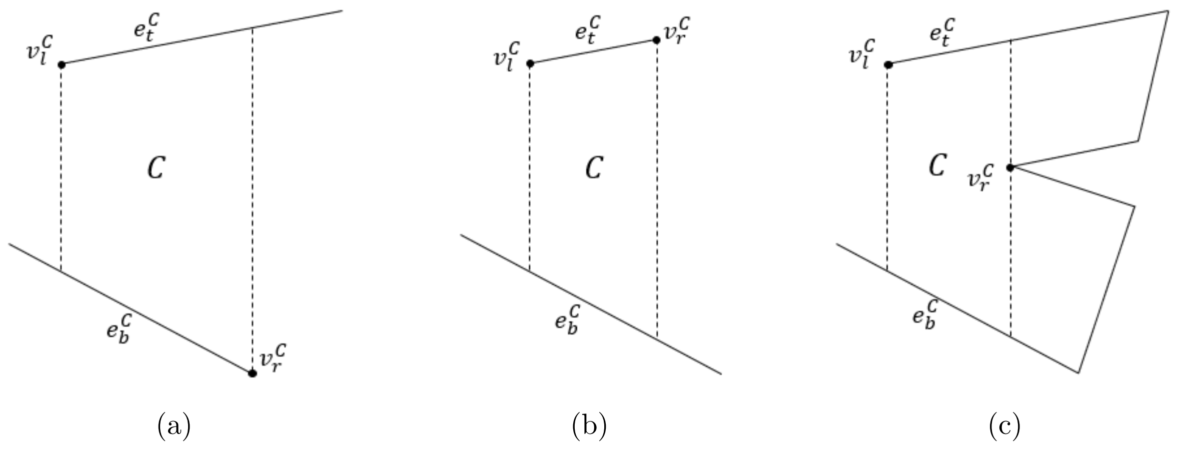


Figure 2: Examples of limiting left and right vertices of a trapezoidal cell C.

We construct these cells by simultaneously performing a radial sweep around all the vertices of the workspace, where each vertex is the origin of a sweeping ray, all rays point at the same direction and are rotated together. For each of these sweeps, we maintain a balanced binary search tree, which contains all the edges of the workspace that the sweeping ray intersects (similar to standard radial plane sweep). An event occurs when one of the rays hits a vertex. We compute all the $O(n^2)$ events in advance and radially sort all of them together (by the angle θ). In addition, each ray stores at all times (angles) the cells on both its sides, allowing us to keep track of all the existing trapezoids of the decomposition at the current orientation. During the handling of an event, where two vertices align along the sweeping ray (its origin and another vertex), if they are visible from one another, namely the open line segment connecting them lies in the interior of the workspace, cells should be created or terminated.

There are two types of events along the radial sweep:

- Two vertices of the same edge are incident to a rotating ray. See Figure 3 for an illustration. The triangular cell that contained both vertices on its boundary is "squeezed" and terminated, and a new triangular cell is created. The cell sharing an artificial edge with the triangular cell also terminates and a new one is created due to change of one of the limiting vertices. In total, two cells are terminated and two are created.

- Two vertices of two different edges are incident to a rotating ray. See Figure 4 for an illustration. The middle cell is "squeezed" and terminated, and a new middle cell is created. The cells from both sides of the connecting ray terminate and new ones are created due to change in the limiting vertices. In total, three cells are terminated and three are newly created.

In both event types, the top and bottom edges and left and right vertices are known locally from the information maintained by the rays or the terminated cells. During handling of an event we terminate and create a constant number of cells. As there are $O(n^2)$ events, there are also $O(n^2)$ cells in total.

To start the process, trapezoidal decomposition in the horizontal direction (namely, with $\theta = 0$), is performed, requiring $O(n \log n)$ time and starting $\Theta(n)$ cells. We let θ vary in the range $[0, 2\pi)$. Notice that the trapezoidal decomposition at $\theta = 0$ artificially cuts cells into two; this however does not affect the analysis that follows or the asymptotic resources required by the algorithm. In total, we construct these $O(n^2)$ cells and their angle intervals in $O(n^2 \log n)$ time, while using $O(n^2)$ space.

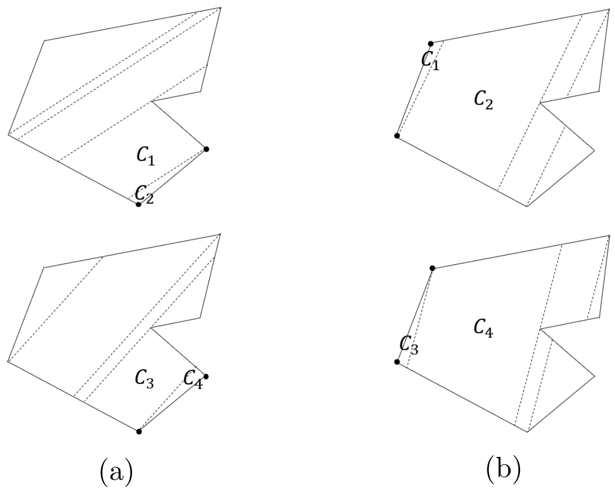


Figure 3: Type I events: two vertices of the same edge align along the rotating ray. Rays are rotating counterclockwise. The event vertices are drawn as small discs. Each column depicts one example, before (top) and after (bottom) the event.

2.2 Storing the Cells for Efficient Placement Retrieval

We wish to sort the cells in the decomposition of the C-space of the sensor such that given a distance measurement d, we can efficiently collect all the cells that have non-empty set of candidate placements for this measurement. To this end we define the maximal opening $O^C_{\max}$ of a cell C, which is the maximal-length θ-oriented segment that can be placed in a θ-cross-section of a cell C, over all $\theta \in \Theta^C$. We define the maximal opening of a cell more formally as follows. Let $e^C_t(\theta)$ be the top edge of the trapezoidal cross-section of the cell C at angle θ. Define the size of the opening at angle θ at x of the top edge, between the top and bottom edges of a cell by $O^C(\theta, x)$, namely the length of the intersection of the line with slope θ through the point $e^C_t(x)$ on the top edge of the cell, with the trapezoidal cross-section of C at angle θ. See Figure 5 for an illustration. Next, let $O^C(\theta) = \max_{\theta \in \Theta^C} \max_{x \in e^C_t} O^C(\theta, x)$. Finally the maximum opening of the cell is defined to be $O^C_{\max} = \max_{\theta \in \Theta^C} O^C(\theta)$.

In a preprocessing phase, we calculate $O^C_{\max}$ for each cell, and keep the cells in an array, where they are sorted by this value. Given a query measurement d, the cells that contain potential placements for this reading are exactly the cells for which $d \leq O^C_{\max}$. We can find these cells using a binary search on the sorted cells, resulting in $O(\log n + k)$ query time, where k is the number of cells that contain workspace points of the desired answer. The calculation of the maximum opening takes constant time per cell. Since there are $O(n^2)$ cells overall, the entire preprocessing requires $O(n^2 \log n)$ time and $O(n^2)$ space, which is asymptotically subsumed by the construction of the decomposition (Section 2.1).

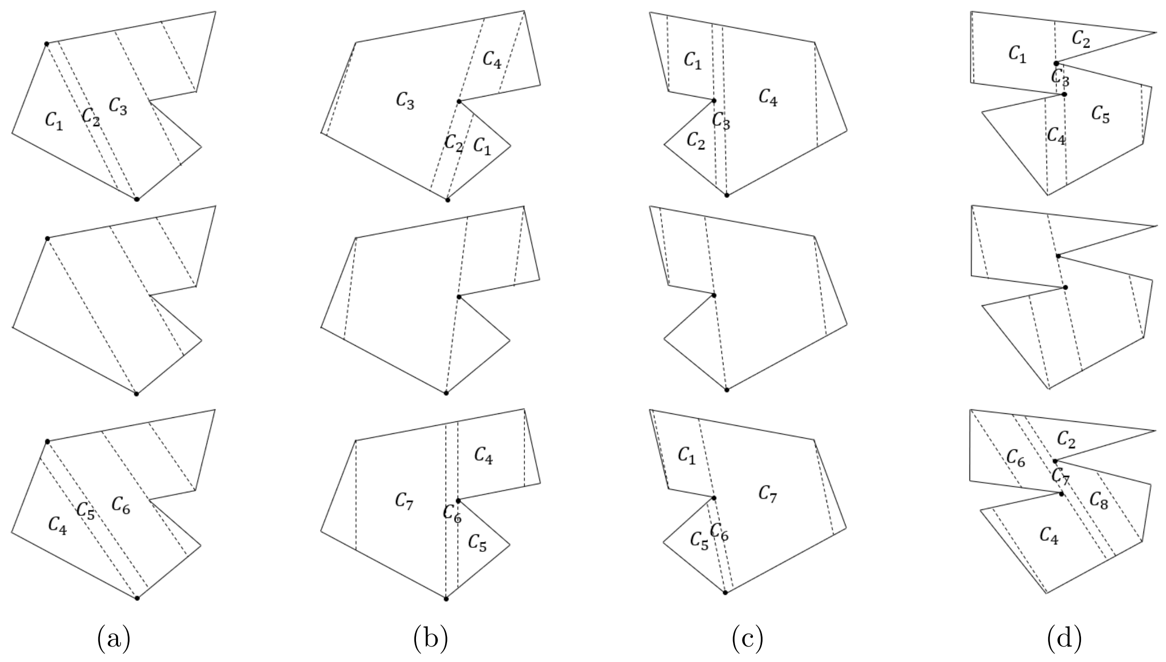


Figure 4: Type II events: two vertices of two distinct edges align along the rotating ray. Rays are rotating counterclockwise. The event vertices are drawn as small discs. Each column depicts one example, before (top), during (middle), and after (bottom) the event.

3 Sensor Positions for a Fixed Reading

Given a sensor reading d, we concern ourselves in the section with the shape and complexity of the locus of sensor *positions* that result in such a reading. In Section 3.1 we show how to compute for each three-dimensional cell C of the decomposition, the region $\Psi_d(C)$ of positions where the sensor could be, namely $\Psi_d(C)$ is the projection onto the xy-plane of C_d. Then, in Section 3.2 we discuss the presentation of the union of the regions $\Psi_d(C)$ over all cells C in the decomposition.

3.1 Sensor Positions for a Single Cell

Each cell C prevails through an interval of θ values, calculated during the sweep—we denote this interval by Θ^C. Given a cell C and an angle $\theta \in \Theta^C$, we can easily construct the trapezoid that is the cross-section of C at θ. The top left and top right vertices (which are not necessarily vertices of the workspace) of the trapezoid both lie on e_t^C. We can compute their x values as a function of θ, denoted $x_{tl}^C(\theta), x_{tr}^C(\theta)$.

Let $p_l^C(\theta)$ denote the position of the sensor, where a distance measure d at angle θ will hit the left endpoint of e_t^C , while ignoring other obstacles in the workspace, that is $(x_{tl}^C(\theta) - d\cos\theta, e_t^C(x_{tl}^C(\theta)) - d\sin\theta)$. Similarly denote by $p_r^C(\theta)$ the placement of the sensor that will result in hitting the right endpoint of e_t^C. Recall that $\Psi_d(C)$ denotes the region $\{(x,y)|(x,y,\theta) \in C$ and $h(x,y,\theta) = d\}$, namely the positions of the sensor in the workspace, where the sensor configuration is in C and there is a sensor reading d. We wish to trace the curves drawn by $p_l^C(\theta)$ and $p_r^C(\theta)$ as θ varies in the range Θ^C. These curves are portions of the boundary of the region $\Psi_d(C)$. The type of the curve that the function $p_l^C(\theta)$ (respectively, $p_r^C(\theta)$) draws, depends on v_l^C (respectively, v_r^C), the vertex defining the left (respectively, right) wall of the cell. We describe it here for v_l^C. The

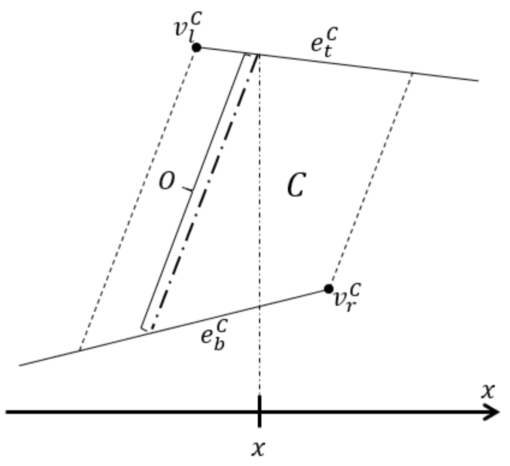


Figure 5: The opening size O as a function of x and θ.

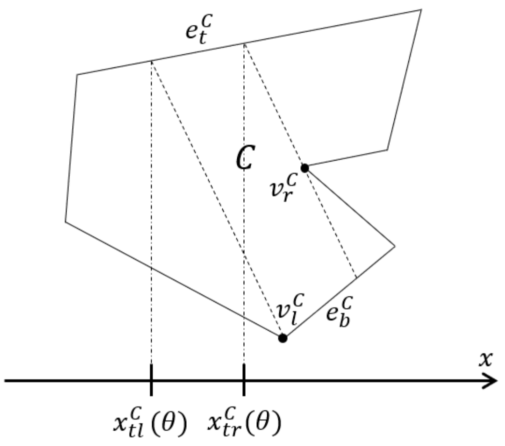


Figure 6: The x values of the left and the right vertices of a cell's ceiling.

description for v_r^C is analogous.

- The vertex v_l^C defining the left wall of the trapezoid/cell lies on e_t^C. See Figure 8a(top) for an illustration. The endpoint v_l^C is fixed (i.e., independent of θ), and the curve is a circular arc of radius d around v_l^C: $v_l^C - (d\cos\theta, d\sin\theta)$.

- The vertex v_l^C defining the left wall of the trapezoid/cell does *not* lie on e_t^C. See Figure 8c(top) for an illustration. The curve traced by $p_l^C(\theta)$ is a conchoid of Nicomedes; see Appendix D for details.

For any angle θ a cell C exists in, all points on the line $(p_l^C(\theta), p_r^C(\theta))$ are points the robot can be and measure d at e_t^C (ignoring other obstacles). After we characterized the types of the curves of $p_l^C(\theta), p_r^C(\theta)$, this gives us an exact description of the area containing all the points the robot might be in this cell. If the cell angle interval contains the angle perpendicular to the top edge, we divide into two sub intervals, one with greater angles than the perpendicular one and other with smaller ones. This splitting is required to create a valid walls for the 2D area representation. For each angle interval, we calculate the two curves $p_l^C(\theta), p_r^C(\theta)$ which gives us a 2D area in the workspace the robot might

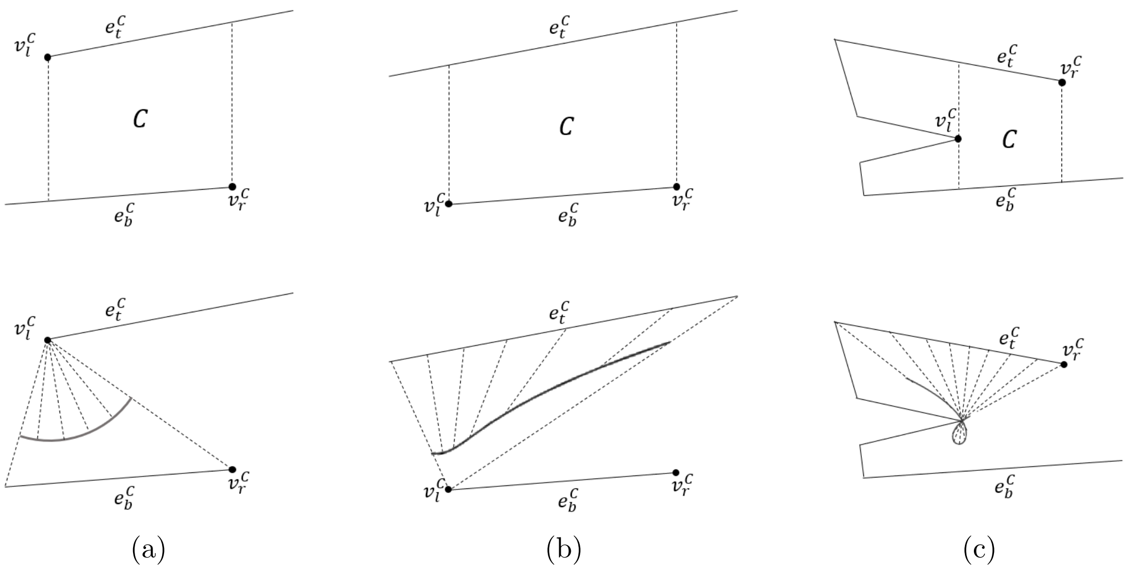


Figure 7: (a) Type I curve: limiting vertex lays on cell ceiling, curve is a simple arc. (b)(c) Type II curve: limiting vertex doesn't lay on cell ceiling, curve is conchoid of Nicomedes.

be in. The only obstacle we need to consider is the bottom edge, and we can simply intersect the result with the top half plane of the bottom edge to get the final result of the cell.

3.2 The Overall Region of Potential Sensor Positions for a Fixed Reading

Given a sensor reading d, we retrieve all the cells with non-empty potential placements using the sorted list of Section 2.2. For each such cell C we compute the region $\Psi_d(C)$. This in itself can serve as an answer to a query, in particular, when we wish to further process the regions using additional information about the potential placement of the sensor.

Alternatively, we may wish to return the (two-dimensional) union of the regions $\Psi_d(C)$, for all the cells that we retrieved. Each region $\Psi_d(C)$ has constant descriptive complexity, and if we retrieved k such regions, then their union, can be easily computed in time $O((k+m)\log(k+m))$, where m is the complexity of the union. In the worst case $m = O(k^2)$. It might be the case that special properties of the regions $\Psi_d(C)$ and their juxtaposition could be used to show that the complexity of the union is $o(k^2)$. We leave this as an open problem for further research.

4 Two Antipodal Distance Measurements

We now turn to the case where the sensor takes two different distance measurement from the same point and in opposing directions. We obtain one measurement d_1 in (unknown) direction θ_1 and a second measurement d_2 in direction $\theta_1 + \pi$. A nice property of this setting is that the two distance measurements are taken to two *distinct* edges of the workspace.

We apply the same decomposition of the sensor C-space as above (Section 2), where the orientation θ is the orientation of the first measurement. For a fixed orientation

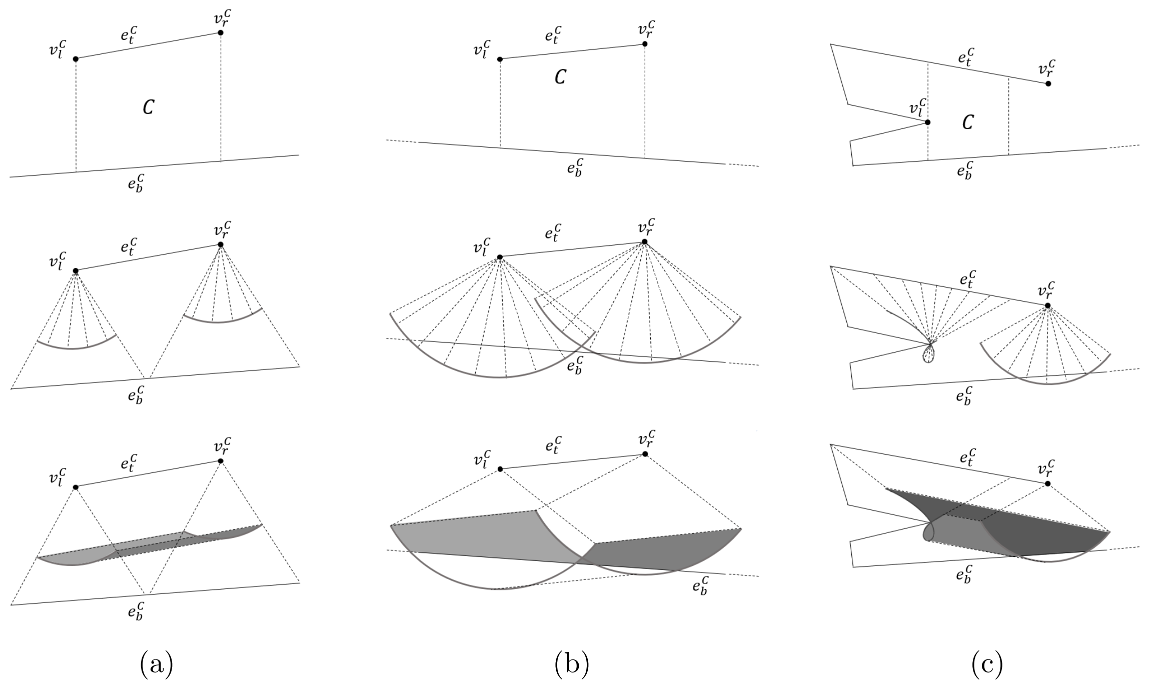


Figure 8: (a) Type I curve: limiting vertex lays on cell ceiling, curve is a simple arc. (b)(c) Type II curve: limiting vertex doesn't lay on cell ceiling, curve is conchoid of Nicomedes.

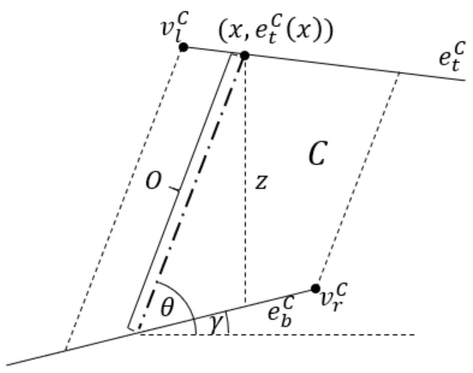
θ, the cross-section of a cell C contains one potential sensor placement (assuming the top and bottom edges of the cell are not parallel), and we can calculate it by solving $O^C(\theta, x) = d_1 + d_2$, and extracting the point along the segment obtaining this opening that is at distance d_1 from the top edge (and hence at distance d_2 from the bottom edge).

For each such cell C we compute the locus $\Gamma(C)$ of potential sensor placements, which is a curve parameterized by the orientation of the first measurement.

In order to store the cells of the decomposition so as to be able to efficiently retrieve the cells C with non-empty loci $\Gamma(c)$, we now need to also compute the minimum opening of a cell $O^C_{\min}$, which is defined analogously to $O^C_{\max}$. Given a pair of antipodal measurements (d_1, d_2), a cell C has a non-empty set of loci if and only if $O^C_{\min} \leq d_1 + d_2 \leq O^C_{\max}$. Each cell is associated with an interval $[O^C_{\min}, O^C_{\max}]$, and we construct an interval tree [1, Chapter 10] over these $O(n^2)$ intervals. Given the measurements (d_1, d_2), we extract all the intervals that contain the value $d_1 + d_2$, and their corresponding cells contribute potential placements to the final answer.

The loci $\Gamma(C)$ constitute an arc of an ellipse with its center at the (possibly imaginary) intersection of the lines supporting the top and bottom edges of the cell. After prepossessing at $O(n^2 \log n)$ time and using $O(n^2)$ space, a query can be answered in $O(\log n + k)$ time, where k is the number of cells C with non-empty $\Gamma(C)$.

A Calculating $O^C(\theta, x)$



We can look at the triangle constructed from O, z, e_b^C. The angle opposite of e_b^C is $\frac{\pi}{2} - \theta$, the angle opposite of z is $\theta - \gamma$ and the angle opposite of O is $\frac{\pi}{2} + \gamma$. Using the law of sines:

$$O = \frac{z \sin(\frac{\pi}{2} + \gamma)}{\sin(\theta - \gamma)} = \frac{z \cos\gamma}{\sin(\theta - \gamma)} = \frac{z \cos\gamma}{\sin\theta \cos\gamma - \cos\theta \sin\gamma}$$

$$= \frac{z \frac{1}{\sqrt{1+m_{e_b}^2}}}{\sin\theta \frac{1}{\sqrt{1+m_{e_b}^2}} - \frac{m_{e_b}}{\sqrt{1+m_{e_b}^2}} \cos\theta} = \frac{z}{\sin\theta - m_{e_b}\cos\theta} = \frac{e_t^C(x) - e_b^C(x)}{\sin\theta - m_{e_b}\cos\theta}$$

$$= \frac{x(m_{e_t} - m_{e_b}) + b_{e_t} - b_{e_b}}{\sin\theta - m_{e_b}\cos\theta}$$

For any fixed angle θ, $O^C(\theta, x)$ is a straight line as function of x, and it's monotonically decreasing iff $m_{e_t} < m_{e_b}$:

$$O^C(\theta, x) = \frac{m_{e_t} - m_{e_b}}{\sin\theta - m_{e_b}\cos\theta} x + \frac{b_{e_t} - b_{e_b}}{\sin\theta - m_{e_b}\cos\theta}$$

We will show that $\sin\theta - m_{e_b}\cos\theta \geq 0$. Each cell C exists in some interval of the values of θ, which is sub interval of $(\tan^{-1} m_{e_b}, \tan^{-1} m_{e_b} + \pi)$. Denote $\theta = \tan^{-1} m_{e_b} + \delta$, $\delta \in (0, \pi)$:

$$\sin\theta - m_{e_b}\cos\theta = \sin(\tan^{-1} m_{e_b} + \delta) - m_{e_b}\cos(\tan^{-1} m_{e_b} + \delta)$$

$$= \sin\tan^{-1} m_{e_b} \cos\delta + \cos\tan^{-1} m_{e_b} \sin\delta - m_{e_b} \cos\tan^{-1} m_{e_b} \cos\delta + m_{e_b} \sin\tan^{-1} m_{e_b} \sin\delta$$

$$= \frac{m_{e_b}}{\sqrt{1+m_{e_b}^2}} \cos\delta + \frac{1}{\sqrt{1+m_{e_b}^2}} \sin\delta - \frac{m_{e_b}}{\sqrt{1+m_{e_b}^2}} \cos\delta + \frac{m_{e_b}^2}{\sqrt{1+m_{e_b}^2}} \sin\delta$$

$$= \frac{1}{\sqrt{1+m_{e_b}^2}} \sin\delta + \frac{m_{e_b}^2}{\sqrt{1+m_{e_b}^2}} \sin\delta = \sqrt{1+m_{e_b}^2} \sin\delta$$

We know $\delta \in (0, \pi)$, which always satisfy $\sin\delta \geq 0$. So $\sin\theta - m_{e_b}\cos\theta \geq 0$, and therefore any fixed angle θ, $O^C(\theta, x)$ is monotonically decreasing iff $m_{e_t} < m_{e_b}$.

For any fixed x, the function $O^C(\theta, x)$ has a minimum at $\theta = \tan^{-1}(-\frac{1}{m_{e_b}})$ and it's monotonically decreasing in θ from the interval beginning until the minimum and monotonically increasing from the minimum until the interval end:

$$sign\left(\frac{\partial O}{\partial \theta}\right) = sign\left(-\frac{\cos\theta + m_{e_b}\sin\theta}{(\sin\theta - m_{e_b}\cos\theta)^2}\right) = -sign(\cos\theta + m_{e_b}\sin\theta)$$

denote $\theta = \tan^{-1} m_{e_b} + \delta$, $\delta \in (0, \pi)$.

$$\cos\theta + m_{e_b}\sin\theta = \cos(\tan^{-1} m_{e_b} + \delta) + m_{e_b}\sin(\tan^{-1} m_{e_b} + \delta)$$
$$= \cos\tan^{-1} m_{e_b}\cos\delta - \sin\tan^{-1} m_{e_b}\sin\delta + m_{e_b}\sin\tan^{-1} m_{e_b}\cos\delta + m_{e_b}\cos\tan^{-1} m_{e_b}\sin\delta$$
$$= \frac{1}{\sqrt{1+m_{e_b}^2}}\cos\delta - \frac{m_{e_b}}{\sqrt{1+m_{e_b}^2}}\sin\delta + \frac{m_{e_b}^2}{\sqrt{1+m_{e_b}^2}}\cos\delta + \frac{m_{e_b}}{\sqrt{1+m_{e_b}^2}}\sin\delta$$
$$= \frac{1+m_{e_b}^2}{\sqrt{1+m_{e_b}^2}}\cos\delta = \sqrt{1+m_{e_b}^2}\cos\delta$$
$$sign\left(\frac{\partial O}{\partial \theta}\right) = sign\left(-\sqrt{1+m_{e_b}^2}\cos\delta\right) = sign(-\cos(\delta))$$

We know $\delta \in (0, \pi)$, so we can determine the sign of the derivative in that range:

$$\delta \in (0, \frac{\pi}{2}) \to sign\left(\frac{\partial O}{\partial \theta}\right) < 0$$

$$\delta \in (\frac{\pi}{2}, \pi) \to sign\left(\frac{\partial O}{\partial \theta}\right) > 0$$

Therefore the function $O^C(\theta, x)$ has a minimum value at $\delta = \frac{\pi}{2} \to \theta = \tan^{-1}(\frac{-1}{m_{e_b}})$, which is intuitive, and the monotonically decreasing/increasing properties mentioned above.

B The Endpoints of a Trapezoid: $x^C_{tl}(\theta), x^C_{tr}(\theta)$

$$e^C_t(x) = m_{e_t}x + b_{e_t}, \quad y = \tan\theta(x - x_{v^C_i}) + y_{v^C_i}$$
$$m_{e_t}x + b_{e_t} = x\tan\theta - x_{v_i}\tan\theta + y_{v_i}$$
$$x(m_{e_t} - \tan\theta) = y_{v_i} - x_{v_i}\tan\theta - b_{e_t}$$
$$x = \frac{y_{v_i} - x_{v_i}\tan\theta - b_{e_t}}{m_{e_t} - \tan\theta}$$
$$x^C_{tl}(\theta) = \frac{y_{v_l} - x_{v_l}\tan\theta - b_{e_t}}{m_{e_t} - \tan\theta} \quad x^C_{tr}(\theta) = \frac{y_{v_r} - x_{v_r}\tan\theta - b_{e_t}}{m_{e_t} - \tan\theta}$$

C Solving $O^C(\theta, x) = d$?

We now turn to solving $O^C(\theta, x) = z$ ($z = d$ in $x_{\Theta^C_2}, x_{\Theta^C_4}$ calculation, $z = d_1 + d_2$ in "Second Measurement Variant"):

$$O^C(\theta, x) = z$$
$$\frac{e^C_t(x) - e^C_b(x)}{\sin\theta - m_{e_b}\cos\theta} = z$$
$$e^C_t(x) - e^C_b(x)) = z(\sin\theta - m_{e_b}\cos\theta)$$

$$m_{e_t}x + b_{e_t} - m_{e_b}x - b_{e_b} = z(\sin\theta - m_{e_b}\cos\theta)$$
$$x = \frac{z(\sin\theta - m_{e_b}\cos\theta) + b_{e_b} - b_{e_t}}{m_{e_t} - m_{e_b}}$$

This result is the x value of the possible measurement on the top edge, to get the possible robot location one should subtract $(d_1\cos\theta, d_1\sin\theta)$.

D Conchoid of Nicomedes representing p_l^C p_r^C

The conchoid of Nicomedes is driven from a fixed point q, a straight line l and a length d, where for every line through q that intersects l, the two points on the line which are d from the intersection are on the conchoid. In our case, q is the limiting vertex, the straight line is the top edge of the cell and d is the query measurements.

Equation for classic conchoid, $q = (0,0)$ $l : x = a$

$$(x-a)^2(x^2+y^2) = d^2x^2$$

Equation for general conchoid, $q = (x_0, y_0)$ $l :$ at angle t, distance a from q:

$$s = \sin t, \quad c = \cos t$$

$$((x-x_0)c-(y-y_0)s-a)^2(((x-x_0)c-(y-y_0)s)^2+((x-x_0)s+(y-y_0)c)^2) = d^2((x-x_0)c-(y-y_0)s)^2$$

In our case, l is the top edge, and its angle is defined by its slope:

$$t = \tan^{-1}(m_{e_t}), \quad \sin\tan^{-1}(m_{e_t}) = \frac{m_{e_t}}{\sqrt{1+m_{e_t}^2}}, \quad \cos\tan^{-1}(m_{e_t}) = \frac{1}{\sqrt{1+m_{e_t}^2}}$$

By using the above identities, we get:

$$\left((x-x_0)\frac{m_{e_t}}{\sqrt{1+m_{e_t}^2}} - (y-y_0)\frac{1}{\sqrt{1+m_{e_t}^2}} - a\right)^2 \left(\left((x-x_0)\frac{m_{e_t}}{\sqrt{1+m_{e_t}^2}} - (y-y_0)\frac{1}{\sqrt{1+m_{e_t}^2}}\right)^2 \right.$$
$$\left. + \left((x-x_0)\frac{1}{\sqrt{1+m_{e_t}^2}} + (y-y_0)\frac{m_{e_t}}{\sqrt{1+m_{e_t}^2}}\right)^2\right) = d^2\left((x-x_0)\frac{m_{e_t}}{\sqrt{1+m_{e_t}^2}} - (y-y_0)\frac{1}{\sqrt{1+m_{e_t}^2}}\right)^2$$

$$\frac{1}{1+m_{e_t}^2}\left((x-x_0)\frac{m_{e_t}}{\sqrt{1+m_{e_t}^2}} - (y-y_0)\frac{1}{\sqrt{1+m_{e_t}^2}} - a\right)^2 \left((m_{e_t}(x-x_0) - (y-y_0))^2\right.$$
$$\left. + ((x-x_0) + m_{e_t}(y-y_0))^2\right) = \frac{1}{1+m_{e_t}^2}d^2(m_{e_t}(x-x_0) - (y-y_0))^2$$

$$\left(\frac{m_{e_t}(x-x_0) - (y-y_0)}{\sqrt{1+m_{e_t}^2}} - a\right)^2 \left(m_{e_t}^2(x-x_0)^2 - 2m_{e_t}(x-x_0)(y-y_0) + (y-y_0)^2\right.$$
$$\left. + (x-x_0)^2 + 2m_{e_t}(x-x_0)(y-y_0) + m_{e_t}^2(y-y_0)^2\right) = d^2(m_{e_t}(x-x_0) - (y-y_0))^2$$

$$\left(\frac{m_{e_t}(x-x_0) - (y-y_0)}{\sqrt{1+m_{e_t}^2}} - a\right)^2 \left((x-x_0)^2 + (y-y_0)^2\right)(1+m_{e_t}^2) = d^2(m_{e_t}(x-x_0) - (y-y_0))^2$$

$$\left(m_{e_t}(x-x_0) - (y-y_0) \pm a\sqrt{1+m_{e_t}^2}\right)^2 \left((x-x_0)^2 + (y-y_0)^2\right) = d^2(m_{e_t}(x-x_0) - (y-y_0))^2$$

The term with $\pm$ sign is determined by relation between q and l. If l is above q, we use the plus sign, if l is below q, we use the minus sign. In total the equation that represent the curve p_l^C within a cell is (can be defined similarly for p_r^C):

$$\left(m_{e_t}\left(x-x_{v_l}\right)-\left(y-y_{v_l}\right)\pm a\sqrt{1+m_{e_t}^2}\right)^2\left(\left(x-x_{v_l}\right)^2+\left(y-y_{v_l}\right)^2\right)=d^2\left(m_{e_t}\left(x-x_{v_l}\right)-\left(y-y_{v_l}\right)\right)^2$$

References

[1] Mark de Berg, Otfried Cheong, Marc J. van Kreveld, and Mark H. Overmars. *Computational Geometry: Algorithms and Applications, 3rd Edition*. Springer, 2008.