# A Portrait Drawing Robot Using a Geometric Graph Approach: Furthest Neighbour Theta-Graphs

Meng Cheng Lau, Jacky Baltes, John Anderson and Stephane Durocher

***Abstract*— We examine the problem of estimating ideal edges joining points in a pixel reduction image for an existing point-to-point portrait drawing humanoid robot, Betty. To solve this line drawing problem we present a modified Theta-graph, called Furthest Neighbour Theta-graph, which we show is computable in $O(n(\log n)/\theta)$ time, where $\theta$ is a fixed angle in the graph's definition. Our results show that the number of edges in the resulting drawing is significantly reduced without degrading the detail of the final output image.**

## I. Introduction

Recent research on humanoid robots has devoted significant effort on developing humanoid robots that can match human behaviour on high-level tasks that require integration of sensing, physical motion and intelligence. Current developments have diverse applications in a wide range of industries, including education, health care, household services, military, entertainment, etc.

A portrait drawing robot requires human-specific skills which are challenging tasks for humanoid robotics. Recent results in the robotics literature include painting robots where various type of systems were implemented [1], [2], [4]. For instance, Gommel et al. [1] implemented an industrial robotic arm, KUKA, to draw in Cartesian space. Lu et al. [2] developed a special purpose robotic arm platform, IRAS, for replicating and creating works of art. However, neither of these robotic systems mimics human-like features successfully. Therefore, in recent years many researchers (e.g., [3], [4], [5], [6], [7], [8]) have tried to develop a robust humanoid robot that could produce pen-and-ink sketches of portraits. This usually requires a tremendous amount of time to complete a task due to complicated motion control and complexity of the input image.

We are working on a portrait drawing humanoid robot, Betty; see Fig. 1. We implemented OpenCV, an open source computer vision library, to perform face recognition and Canny edge detection that computes a line-art portrait that can be mapped to the kinematics of the arm. Currently, Betty draws the portrait using a point-to-point drawing mechanism which maps the points from a pixel-reduction image that consists of a set of points, $P$, as shown in Fig. 2. This article aims to develop a human portrait drawing system that enables Betty to autonomously draw the portrait by implementing a modified Theta-graph, called Furthest Neighbour Theta-graph, to construct a good approximation of the input image by defining edges joining pairs of points in $P$.

M. C. Lau is sponsored by Universiti Kebangsaan Malaysia. mengcheng.lau@gmail.com

J. Baltes, J. Anderson and S. Durocher are with the Department of Computer Science, University of Manitoba, Winnipeg, Manitoba R3T 2N2, Canada. jacky@cs.umanitoba.ca, andersj@cs.umanitoba.ca, durocher@cs.umanitoba.ca

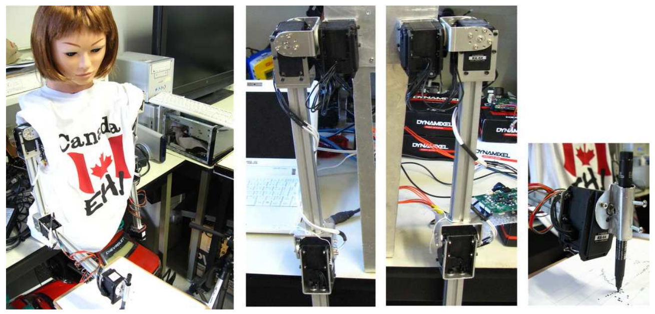


Fig. 1. Betty - the humanoid robot artist

Section II discusses related results on theta-graphs and Yao-graphs. In Sections III and IV, Furthest Neighbour Theta-graph are defined and their implementation is described. Finally, in Sections V and VI, the results of the implementation and possible future research directions are discussed.

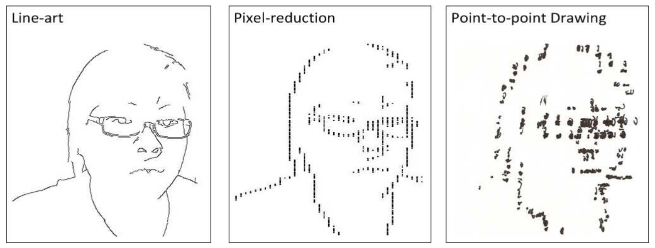


Fig. 2. ***Left***: Line-art Portrait from Canny edge detection algorithm. ***Middle***: Pixel-reduction to reduce the number of pixels. ***Right***: Actual drawing produce by Betty

## II. Background and Related Work

A geometric graph is a weighted graph whose vertex set is a set of points, $P$, in $d$-dimensional Euclidean space, $\mathbb{R}^d$, and the edges of the graph consist of line segments, each of which joins two vertices. The weight of any edge is the Euclidean distance, $L_2$, between its endpoints [9]. Geometric graphs are used to model many practical problems in various fields of computer science and computational geometry.

Theta-graphs [10], [11] and Yao-graphs [12] are popular geometric graphs that appear in the context of navigating graphs [9]. Theta-graphs and Yao-graphs differ in the way the nearest neighbour is defined. In both graphs, every vertex is joined by an edge to each of its nearest neighbours in each cone. In 2D Euclidean space, each cone forms an angle of $\Theta_k = 2\pi/k$, for some fixed $k > 0$. For Yao-graphs ($Y_k$-graphs), the nearest neighbour of $p$ in the cone $C$ is simply a vertex $q \neq p$ in $C$ minimizing the Euclidean distance ($L_2$-distance) between $p$ and $q$. For theta-graphs ($\Theta_k$-graphs), the nearest neighbour of $p$ is the vertex $q \neq p$ whose orthogonal projection onto the bisector of $C$ minimizes the $L_2$-distance to $p$ [9].

Bose et al. introduced a new variant of theta-graph called ordered-$\Theta$-graphs, which are built incrementally by inserting the vertices one by one so that the resulting graph depends on the insertion order [13]. They show that specific insertion orders can produce graphs with desirable properties, including low spanning ratio, logarithmic maximum degree and logarithmic diameter. Bonichon et al. introduced a specific subgraph of the $\Theta_6$-graph defined in $\mathbb{R}^2$, called half-$\Theta_6$-graph, which consist of the even-cone edges of the $\Theta_6$-graph [9]. They show that these graphs are exactly the TD-Delaunay graphs, and are strongly connected to the geodesic embeddings of orthogonal surfaces of coplanar points in 3D Euclidean space.

In this article, we are interested in estimating the ideal edges joining points in the set of points of a pixel reduction image. Therefore we propose a geometric graph called Furthest Neighbour Theta-graphs which are adapted from Theta-graphs [10], [11], [14] and ordered-$\Theta$-graphs [13] which we discuss in Section III. The edges of a Theta-graph are defined by the nearest neighbours of each vertex, which often results in an unrealistic outline for the portrait with high number of discontinued outline edges. Due to a large amount of short edges drawn, a Theta-graph increases the complexity of the portrait outline and its drawing time. In contrast to Theta-graphs, a Furthest Neighbour Theta-graphs produces an appropriate outline for the portrait by connecting its furthest neighbours with reduced edges and low discontinued outline edges as seen in Fig. 8.

## III. Furthest Neighbour Theta-graphs

In this section, the formal definition of Furthest Neighbour Theta-graphs ($\Theta$-graphs) is given and some of its basic properties are established. Let $P$ be a set of $n$ points in the plane and let $\theta$ be an angle such that $k_\theta = 2\pi/\theta$ is a positive integer. For each point $p \in P$, partition the corresponding disc $D$, with radius $r$ into a set of $k_\theta$ cones $C$, each spanning an angle of $\theta$ with apex at $p$; see Fig. 3. Then, add an edge joining $p$ to the vertex in each cone of $C$ whose projection onto the clockwise cone boundary is the furthest $L_2$-distance from $p$. Therefore, for any point set $P$, the Furthest Neighbour Theta-graph $G = (P, E)$ of $P$ has at most $k_\theta n$ edges of $E$ in disc $D$.

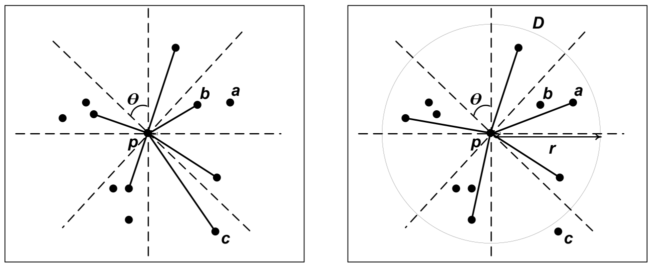


Fig. 3. Comparison between the theta-graph and a Furthest Neighbour $\Theta_8$-graphs with $k_\theta = 8$. ***Left***: Illustration of notations for $\Theta_8$-graphs where the edges are defined by nearest neighbours of $p$, ***Right***:An example of Furthest Neighbour $\Theta_8$-graphs at $p$, where $a$ is the furthest neighbour of $p$ compared to $b$, and $c$ is not bounded by disc $D$

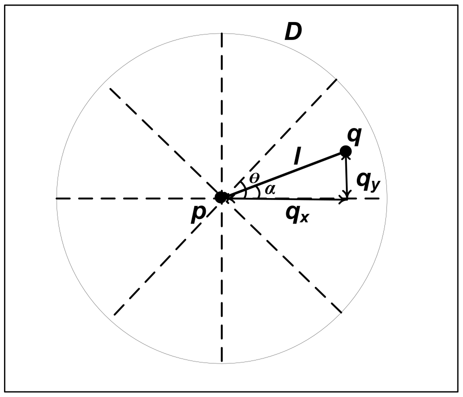


Fig. 4. Finding the furthest neighbour $q$ of $p$ on disc $D$ with the furthest $L_2$-distance $l$ in the first $k_\theta$-cone

Using an algorithm analogous to that described by Bose et al. [13] for constructing ordered $\Theta$-graphs, for any set $P$ of $n$ points in $\mathbb{R}^2$ and any point in disc $D$, the Furthest Neighbour Theta-graph $G$ can be computed in $O(n(\log n)/\theta)$ time. The construction algorithm for Furthest Neighbour Theta-graphs requires finding the furthest neighbours of each point $p \in P$ bounded by a $k_\theta$-cone disc with apex at $p$. We can use $k_\theta$ range trees [15] for each cone. In each tree, every point $q$ of $P$ is stored using a coordinate transformation to $(x, y)$, where $x$ and $y$ correspond to the respective distances between the projections of $p$ and $q$ on the boundaries of the given cone with apex at $p$. See Figs. 3 and 4.

Each vertex requires adding at most $k_\theta$ edges, each of which is determined using one range search. Thus, the graph can be constructed using a series of $k_\theta n$ searches in range trees. Each range tree requires $O(n \log n)$ space and supports construction in $O(n \log n)$ time [16], with $O(\log n)$ update time [17]. Using this implementation of range trees, the above algorithm computes a Furthest Neighbour Theta-graph in $O(n(\log n)/\theta)$ time.

## IV. Implementation

In this article, a practical implementation of furthest neighbour theta graphs is proposed to solve the current portrait sketching problem of our humanoid robot. However the implementation is not limited to a graph algorithm, but it also includes several image processing algorithms with OpenCV such as thresholding, Canny edge detection [18] and edge thinning. The development took place on a Linux platform with Qt Creator and OpenCV library. Qt Creator is a cross-platform integrated development environment (IDE) which supports C++ for Qt program development. Fig. 5 shows the flowchart of the general implementation to generate a sketch-like portrait.

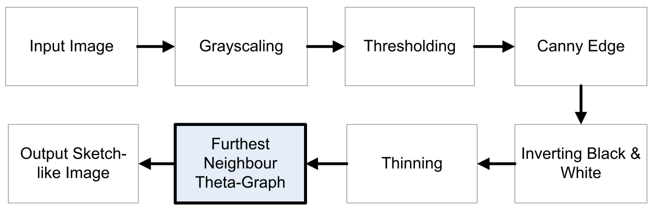


Fig. 5. Overview of the general implementation flowchart

The first step of the implementation is colour conversion from a four-channel RGBA image to a single-channel grayscale image. Next, it generates a binary image from the grayscale image by applying a fixed-level thresholding to remove noise in the output image where pixels with value beyond the threshold are filtered. After thresholding, a Canny edge detector is applied. In this case, it produces a full size image as the input image, but with only black (0) and white (255) pixels, known as single-channel boolean image. The algorithm processes each individual edge candidate pixel into intensity gradients by applying an hysteresis threshold to the pixels, where the higher values are more likely to correspond to edges compared to smaller values. Therefore two thresholds parameters are required: upper and lower thresholds. If a pixel has a gradient higher than the upper threshold, then it is accepted as an edge pixel; if a pixel is less then the lower threshold, it is rejected. If the pixel's gradient is between the thresholds, then it will be accepted only if it is connected to a pixel that is above the high threshold [19].

The next step is to invert the black (0) and white (255) pixels from the output edges by applying bitwise XOR operation on the image array with the *cvXorS()* function. The bitwise XOR is computed with the constant scalar value [19]. In order to reduce the number of pixels to a reasonable number, an image thinning algorithm is deployed with a 3x3 convolution kernel anchored at the middle of the kernel. The function *cvFilter2D()* applies arbitrary linear filter to the image and then interpolates outlier pixel values from the nearest pixels. Its thinning output is illustrated in Fig. 7.

In the final step, the line sketch of a portrait is generated with the furthest neighbour theta-graph algorithm. However, to reduce the number of less significant edges in our image, a dual-disc approach is implemented as illustrated in Fig. 6. Fig. 6 (Left) shows that the initial single-disc approach produces output noise if the neighbours are too close to $p$, which will affect the presentation of the portrait outline. Consequently, Fig. 6 (Right) illustrates a proposed dual-disc approach to solve the problem by only considering the points which fall between the inner and outer discs' boundaries. The results are discussed in Section V.

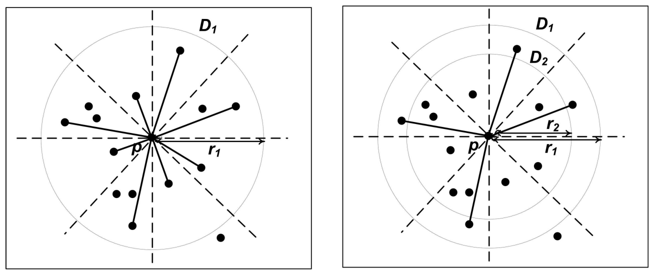


Fig. 6. Implementation of the furthest neighbour theta-graph. ***Left***: The initial proposed single-disc approach. ***Right***: A dual-disc approach to reduce the number of less significant edges.

The algorithm for finding the edges of $E$ at any point of $P$ could be implemented as follows:

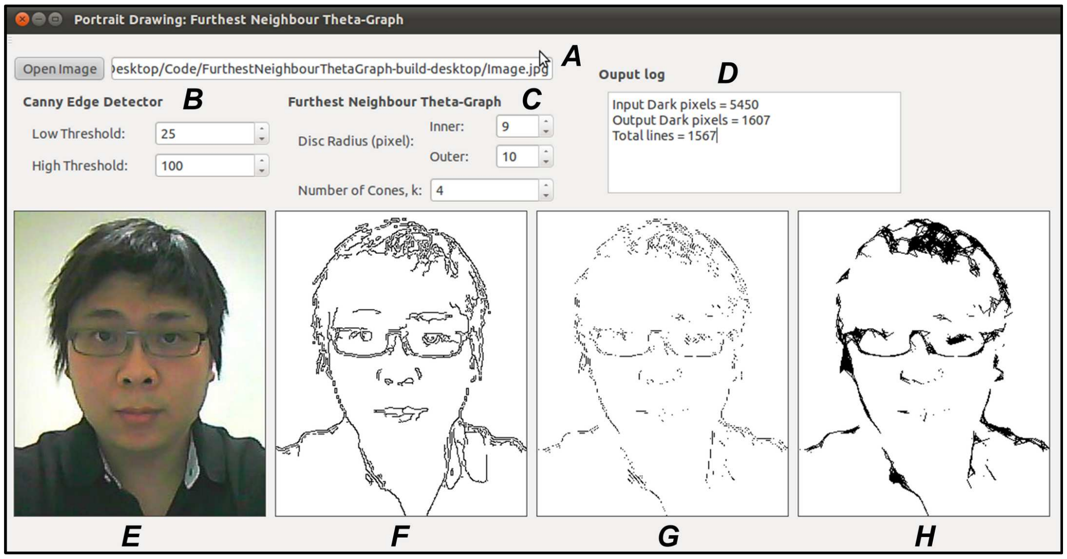
```
1. Read dark pixels into a sorted list
2. For each dark pixel $p$
    a. Superimpose $D_1$ and $D_2$ centred at $p$
    b. For each cone in $k_\theta$ cones
        Search and update the furthest neighbour
        within $D_1$ and $D_2$
    c. Add an edge $e$ to $p$
3. For each edge $e$
    Remove overlap edges if $e_{p,q} = e_{q,p}$ where $p$
    and $q$ denote the end points of the edge
4. Draw edges
```

Fig. 7 shows the implementation of the interactive GUI which computes the furthest neighbour theta-graph to the input portrait. By modifying the parameters of various properties on the graph (e.g, discs' radii and number of cones) a sketch output with fewer edges could be generated.

## V. Results and Discussion

Fig. 8 shows the output of several output sketches generated by different parameters configuration of the furthest neighbour theta-graph. It is clear that the accuracy of its estimated edges is greatly affected by the bounded perimeter of the discs which could be too fine (too many insignificant edges) or too coarse (loss of detail, such as the eyeglasses frame in Fig. 8) to correctly represent the portrait.

The output results are evaluated based on the number of edges produced by the algorithm. Table I shows the output results by varying the radii of the inner and outer discs as well as their respective total number of edges based on various numbers of cones, $k_\theta$. Theses results were obtained from an input image with a 25:100 Canny's thresholds ratio, resulting in 5450 pixels before thinning and 1607 pixels after



esktop/Code/FurthestNeighbourThetaGraph-build-desktop/Image.jpg

Canny Edge Detector

Low Threshold: 25

High Threshold: 100

Furthest Neighbour Theta-Graph

Disc Radius (pixel): Inner: 9 Outer: 10

Number of Cones, k: 4

Ouput log

```
Input Dark pixels = 5450
Output Dark pixels = 1607
Total lines = 1567
```

Fig. 7. Screenshot of GUI. **A**: Open an image. **B**: Canny edge detector thresholds. **C**: Furthest neighbour theta-graph properties. **D**: Output log for processing results. **E**: Input image. **F**: Edge detection by Canny algorithm. **G**: Thinning image after morphological operation. **H**: Output of furthest neighbour theta-graph

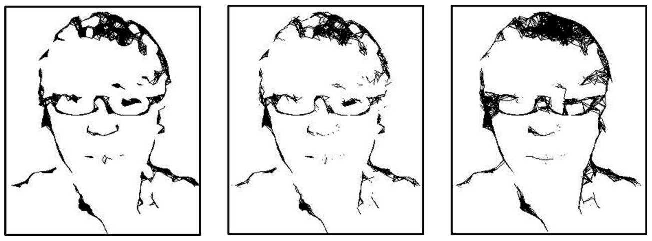


Fig. 8. Different output sketches based of discs' radii ratio $r_{inner}$:$r_{outer}$. ***Left***: 0:10, ***middle***: 9:10 and ***right*** 14:15 with four cones, $k_\theta = 4$

thinning. This suggests that the number of edges could be reduced according to the inner and outer radii ratio, where a smaller range of disc perimeters will produce fewer edges. As seen in Table I, the number of edges of 2, 4, 6 and 8 cones configurations are reduced by 63%, 62%, 69% and 73% respectively. However, due to the limitation of drawing output assessment; the final selection of threshold levels is based on visual evaluation of the output images. For example, as the middle portrait in Fig. 8, we found that setting $k_\theta = 4$ and a discs' radii ratio, $r_{inner}$:$r_{outer}$, of 9:10 produced reasonable output that faithfully captures and presents the portrait's detail, e.g., glasses frame.

The comparison chart shown in Fig. 9 illustrates the effectiveness of the furthest neighbour theta-graph at reducing the total number of output edges with different values of $k_\theta$ and discs' radii. By removing insignificant edges from the sketches, the total number of drawing motions required by Betty is reduced, so it can complete the sketching in less time. Therefore, it improves the portrait outline of our older point-to-point portrait drawing method.

TABLE I

OUTPUT EDGES OF $k_\theta$-CONE COMPARISON BASED ON CANNY'S THREAHOLD- 25:100, INPUT PIXELS- 5450 AND OUTPUT PIXELS- 1607

| $k_\theta$-cone | Inner radius (Pixels) | Outer radius (Pixels) | Output edges |
|---|---|---|---|
| 2 | 0 | 10 | 3427 |
| 2 | 5 | 10 | 2096 |
| 2 | 9 | 10 | 1264 |
| 4 | 0 | 10 | 4137 |
| 4 | 5 | 10 | 3323 |
| 4 | 9 | 10 | 1567 |
| 4 | 14 | 15 | 1889 |
| 6 | 0 | 10 | 5519 |
| 6 | 5 | 10 | 4149 |
| 6 | 9 | 10 | 1709 |
| 8 | 0 | 10 | 6718 |
| 8 | 5 | 10 | 4785 |
| 8 | 9 | 10 | 1786 |

## VI. CONCLUSION AND FUTURE RESEARCH

This article has presented and implemented a modified Theta-graph called Furthest Neighbour Theta-graph to solve its line drawing problem in $O(k_\theta n \log n)$ time. Our algorithm produces a sketch-like portrait drawing, which could significantly reduce the number of pixels drawn by a robot such as Betty, while retaining the important details of the input image. As future work, we will examine extensions of this approach in four directions. First, we wish to take other algorithms such as Bresenham's line algorithm into account, particularly to form a closer approximation to a straight edge between two given points. Second, we will

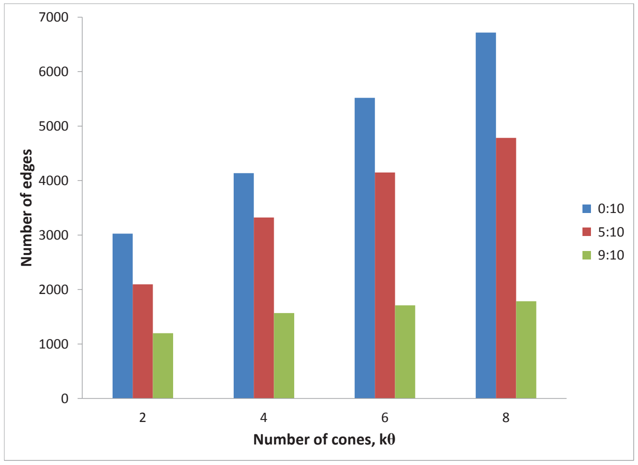


Fig. 9. Comparison of number of edges in the output images

consider the application of other convolution kernels or filters in the thinning algorithm to generate better estimates of the input image with fewer pixels. Then, the application of machine learning techniques will be considered to automatically select and tune the existing parameters instead of manually specifying them. Finally, we want to reduce the construction time by considering a hybrid of other geometric graphs, such as unit disc graphs, while still producing reasonable sketches of portraits.

## References

[1] M. H. M. Gommel and J. Zappe, "autoportrait project: Portrait drawings with a robotic arm," 2004. Available at: http://www.robotlab.de/auto/portrait.htm.

[2] Y. Lu, J. H. M. Lam, and Y. Yam, "Preliminary study on vision-based pen-and-ink drawing by a robotic manipulator," in *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, (Singapore), pp. 578–583, IEEE, July 2009.

[3] M. C. Lau and J. Baltes, "The real-time embedded system for a humanoid: Betty," in *Proceedings of the 13th FIRA Robot World Congress*, vol. 103 of *Communications in Computer and Information Science*, (Bangalore, India), pp. 122–129, Springer-Verlag Berlin Heidelberg, September 2010.

[4] S. Calinon, J. Epiney, and A. Billard, "A humanoid robot drawing human portraits," in *Proceedings of the IEEE-RAS International Conference on Humanoid Robots (HUMANOID 2005)*, (Tsukuba, Japan), IEEE-RAS, December 2005.

[5] C. Y. Lin, L. W. Chuang, and T. T. Mac, "Human portrait generation system for robot arm drawing," in *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, (Singapore), pp. 1757–1762, IEEE, July 2009.

[6] T. Olsson, J. Bengtsson, R. Johansson, and H. Malm, "Force control and visual servoing using planar surface identification," in *Proceedings of the 2002 IEEE International Conference on Robotics & Automation*, (Washington, DC), pp. 4211–4216, IEEE, May 2002.

[7] S. Kudoh, K. Ogawara, M. Ruchanurucks, and K. Ikeuchi, "Painting robot with multi-fingered hands and stereo vision," *Robotics and Autonomous Systems*, vol. 57, pp. 279–288, 2009.

[8] A. Srikaew, M. E. Cambron, S. Northrup, R. A. Peters, II, R. A. P. Ii, D. M. Wilkes, and K. Kawamura, "Humanoid drawing robot," in *In Proceedings of the IASTED International Conference on Robotics and Manufacturing*, 1998.

[9] N. Bonichon, C. Gavoille, N. Hanusse, and D. Ilcinkas, "Connections between theta-graphs, delaunay triangulations, and orthogonal surfaces," in *Graph Theoretic Concepts in Computer Science* (D. Thilikos, ed.), vol. 6410 of *Lecture Notes in Computer Science*, pp. 266–278, Springer Berlin / Heidelberg, 2010.

[10] K. Clarkson, "Approximation algorithms for shortest path motion planning," in *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, STOC '87, (New York, NY, USA), pp. 56–65, ACM, 1987.

[11] J. M. Keil, "Approximating the complete euclidean graph," in *No. 318 on SWAT 88: 1st Scandinavian workshop on algorithm theory*, (London, UK), pp. 208–213, Springer-Verlag, 1988.

[12] A. C.-C. Yao, "On constructing minimum spanning trees in $k$-dimensional spaces and related problems," *SIAM Journal on Computing*, vol. 11, no. 4, pp. 721–736, 1982.

[13] P. Bose, J. Gudmundsson, and P. Morin, "Ordered theta graphs," *Comput. Geom. Theory Appl.*, vol. 28, pp. 11–18, May 2004.

[14] J. Gudmundsson, G. Narasimhan, and M. Smid, "Geometric spanners," *Encyclopedia of Algorithms*, pp. 360–364, 2008.

[15] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.

[16] J. L. Bentley, "Multidimensional Divide-and-Conque," *Communications of the ACM*, vol. 17, pp. 703–724, 1980.

[17] K. Mehlhorn and S. Näher, "Dynamic fractional cascading," *ALGORITHMICA*, vol. 5, no. 1, pp. 215–241, 1990.

[18] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, pp. 679–698, November 1986.

[19] B. Gary and K. Adrian, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 3rd. ed., 2008.