

POLITECNICO DI TORINO

Master degree course in Mechatronic Engineering

Master Degree Thesis

**Automatic detaching methodology of metal
sheet components after laser cutting process**



SUPERVISORS

Prof. Paolo Chiabert
PhD. Khurshid Aliev

CANDIDATE

Sunnatilla Samatov

October 2020

Abstract

This thesis is a part of industry project aimed to automate detaching and sorting metal sheet components after laser cut. Geometrical centroids of components can be outside of their shapes; it may cause problems for robot that is removing them from metal sheet. Focus of this work is to find a suitable grasping point that is located within the shape. Key parameters such as center of mass, principal inertia axes and medial axis are analyzed. Practical solutions are investigated based on key parameters. Maximum inscribed circle inside polygon using Voronoi diagram methodology is chosen to solve grasping point problem. Solution is implemented in Matlab and results are tested on the Universal Robot 3.

Contents

Abstract	3
1. Introduction	9
1.1 Toward Industry 4.0	9
1.2 Sheet metal cutting line	9
1.3 Detaching and sorting problems	10
1.4 Focus of this thesis	10
2. Existing solutions for grasping point	11
2.1 Pole of Inaccessibility	11
2.2 Polylabel	12
2.3 CAUV	13
2.4 MICGIS	13
2.5 Maximum inscribed circle	15
3. Possible key parameters for grasping point of samples without micro-joint	15
3.1 Center of mass	15
3.1.1 COM and Centroid	17
3.1.2 COM of complex shapes	19
3.1.3 The center of gravity	21
3.2 Inertia Moments	21
3.2.1 Area moments of inertia	21
3.2.2 Mass moment of inertia	25
3.2.3 The Search for Principal Axes as Eigenvalue Problem	27
3.3 Medial Axis and Its Computation Methods	28
3.3.1 Thinning algorithms	29
3.3.2 Distance Field Computations	29
3.3.3 Algebraic Methods	30
3.3.4 Surface sampling approaches	30
3.3.5 Medial Axis construction through Voronoi diagram	30
4. Applications of key parameters and methods	30
4.1 Center of mass computation	30
4.1.1 Manual computation of center of mass	31
4.1.2 CAD computation of center of mass	40
4.1.3 Obtained center of mass results	41
4.2 Grasping point along principal axes	43
4.3 MIC implementation using Voronoi diagram in Matlab	44

3.2.1 MIC algorithm steps.....	45
4.3.2 Obtained MIC results	46
5.Implementation of MIC results to Universal Robot 3 (UR3).....	48
5.1 Overview of UR3.	48
5.1.1 Limitations of UR3.....	48
5.1.2 UR 3 programming.....	48
5.2 Workflow of MIC from Matlab to UR3	49
5.2.1 Testing MIC results on UR3	50
6.Applications of MIC results for samples with micro-joints	50
7. Conclusions	51
8. Bibliography.....	53
9. Appendices	55

List of figure

Figure 1. Laser cutting line	9
Figure 2. Manual detaching and detaching with instrument.....	10
Figure 3. Pole of Inaccessibility sample.....	11
Figure 4. Polylabel sample.....	12
Figure 5. CAUV sample	13
Figure 6. MICGIS sample	14
Figure 7. MIC using Voronoi diagram sample	15
Figure 8. Plumb line method sample	19
Figure 9. Area decomposition sample.....	20
Figure 10. Area moments of Inertia sample.....	21
Figure 11. Second Moment of Area of Composite Shape.....	24
Figure 12. Mass Moment of Inertia sample	26
Figure 13. Principal axes in common shapes	27
Figure 14. Principal axes in 3D	27
Figure 15. Straight skeleton of polygon	29
Figure 16. Work piece sketch in CAD	31
Figure 17. Sample 1	32
Figure 18. Sample 2	32
Figure 19. Sample 3	33
Figure 20. Sample 4	35
Figure 21. Sample 5	36
Figure 22. Sample 6	36
Figure 23. Sample 7	37
Figure 24. Sample 8	38
Figure 25. Extruded CAD version of work piece.....	40
Figure 26. Principal inertial axes of sample 5 and sample 6.	43
Figure 27. MIC simulation result in Matlab.....	46
Figure 28. UR3	48
Figure 29. Workflow diagram.....	49

List of table

Table 1. Centroids of common shapes	19
Table 2. Origin of reference frame is located at the centroid of shape.....	22
Table 3. Origin of reference frame is located different from centroid	24
Table 4. Manual and Solidworks COM results	42
Table 5. MIC Centroids	47

Abstract

This thesis is a part of industry project aimed to automate detaching and sorting metal sheet components after laser cut. Geometrical centroids of components can be outside of their shapes; it may cause problems for robot that is removing them from metal sheet. Focus of this work is to find a suitable grasping point that is located within the shape. Key parameters such as center of mass, principal inertia axes and medial axis are analyzed. Practical solutions are investigated based on key parameters. Maximum inscribed circle inside polygon using Voronoi diagram methodology is chosen to solve grasping point problem. Solution is implemented in Matlab and results are tested on the Universal Robot 3.

1.Introduction

1.1 Toward Industry 4.0

Current era of technologies is moving to Fourth Industrial Revolution (Industry 4.0). Specially, traditional manufacturing and its operations in factories are trying to make high level of automation that allows machine to machine communication (M2M). This leads to change physical production into advanced manufacturing in the forms of additive manufacturing, advanced materials, smart machines and other technologies. Main aims of companies transition to Industry 4.0 are to be faster, more efficient, less human interaction, higher quality, less waste disposal and more flexible in manufacturing.

1.2 Sheet metal cutting line

Small and medium enterprises (SME) related to sheet metal field are trying to follow path toward Industry 4.0 in order to achieve less human interactions and fast production. Our interested area is the level of automation of sheet metal cutting line, especially laser cutting line in manufacturing.

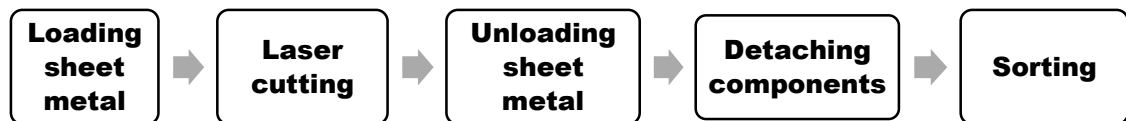


Figure 1. Laser cutting line

Processes of laser cutting line are shown in the figure 1.

- Loading sheet metal are done manually and automatically. One or two operators can feed laser cutting machine with sheet metal depending on its size in manual mode. Industrial robots and conveyor systems are used frequently in automated mode.
- According to laser beam movement, three types of laser cutting machines are used in industry. They are classified into three such as moving material, flying optics and hybrid. The most common one is flying optics, that are automated under CNC parameter, in sheet metal cutting. CNC programming allows cutting beam to move fast and accurately.
- Unloading process is also performed manually and automatically as in loading process.
- Detaching process has two forms without, and with micro-joints. Only detaching of some reasonable large cross sectional components without micro-joints are automated while the rest ones are performed manually or with detaching instruments.
- Sorting is directly linked with detaching process. If detaching is automated, sorting will be the same. In other cases, process is done manually by operator.

1.3 Detaching and sorting problems

It can be met a lot of human operations in SMEs. One of the cases is an operator detaching metal components from the metal sheet after a laser cutting. Metal components may have with or without micro joints after a laser-cut of metal sheet.



Figure 2. Manual detaching and detaching with instrument

Human operator detaches metal components by hand and by using a rubber mallet or a vibration hammer with air depending on existence and non-existence of micro joints in the figure 2. Repetitive and uninteresting tasks, constrained body posture may cause mental and physical stress to human operator. Sometimes this may lead a deformation of work pieces. On the other hand, fiber laser technology provides high processing speed in flat sheet laser cutting. Removal and sorting of components causes bottlenecks in production.

1.4 Focus of this thesis

Solutions to problem points mentioned above can be integrating collaborative and mobile robots into workplaces. Automation of repetitive and boring tasks, improvement of productivity, high precision can be achieved by application of those robots and cobots. Cobots are (collaborative robots) robots that interact with human without safety barriers, and they can assist to human operator as supporting tool or co-worker. Knowing such potentials of cobot, it can be implemented extract laser cut components from sheet metal in workplace. It is expected to investigate possible scenarios theoretically and practically:

- Automation of detaching components without micro joints

- Automation of detaching components with micro joints

Main problem of detaching is to identify an appropriate grasping point for each metal sheet component. It could be the case that center of mass is not within the shape. Main focus of this thesis is to find a reasonable solution for grasping point and to test the solution on the real cobot.

2. Existing solutions for grasping point

It can be seen more often centroid is outside the shape. It is very crucial a centroid being within polygon for grasping purposes, otherwise there is no physical meaning of centroid. This problem is known as Maximum Inscribed Circle (MIC) within polygon. MIC is used wide range of fields such as cartography, landscape planning, urban planning, geology, forestry, agriculture, medicine, astronomy and so on.

2.1 Pole of Inaccessibility

MIC is defined as Pole of Inaccessibility which is related to geography. The aim of pole of inaccessibility is finding the remotest places on the earth. Algorithm is introduced by [1], it solves the problem iteratively with arbitrary precision.

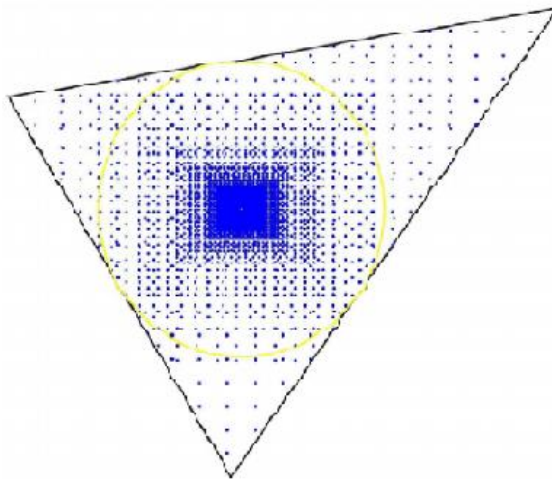


Figure 3. Pole of Inaccessibility sample

This algorithm works by following steps:

1. Define R region as rectilinear between (x_{\min}, x_{\max}) latitude and (y_{\min}, y_{\max}) longitude ranges.
2. Divide R into 21 points (it means dividing the height and width by 20). The number of points is arbitrary, in this paper 21 points are chosen.

3. Cut off any points that are not inside polygon
4. Find the point that is the most distant from any point on the edge
5. Step 2 is repeated once that point defined as a new R with smaller intervals. The paper decreases R by square root of 2.

While evaluating a point whether inside or outside of polygon, cast a ray solution is implemented. It is assigned as an odd and even number, once point is on the edge or within polygon, and is outside from polygon, respectively. Algorithm runs until the search gets a satisfactory precision we want. Algorithm does not always provide a good solution for any polygon since it depends on relatively well-shaped polygons. Another issue could be its slow speed on large polygons because of a lot of points to check. For every blue point in the figure 3, it is required iteration through all polygon points.

2.2 Polylabel

Polylabel [2] is a fast algorithm for finding the pole of inaccessibility. Solution is based on quadtree method. The main concept of quadtree is to subdivide recursively two-dimensional region into four quadrants.

Algorithm working steps:

1. Covering the polygon fully with initial square cells, then calculating distance from the center of each cell to the outer polygon (negative value is used if point is outside of polygon)
2. Putting the cells into a priority queue sorted by the maximum potential distance a point inside a cell, computed as a sum of the distance from the center and cell radius.
3. Calculating the distance from the centroid of polygon and accept it as the first candidate point.
4. Cell's best distance is selected by priority queue. If potential cell has better solution, it is divided into four quadrants and are put in the queue.

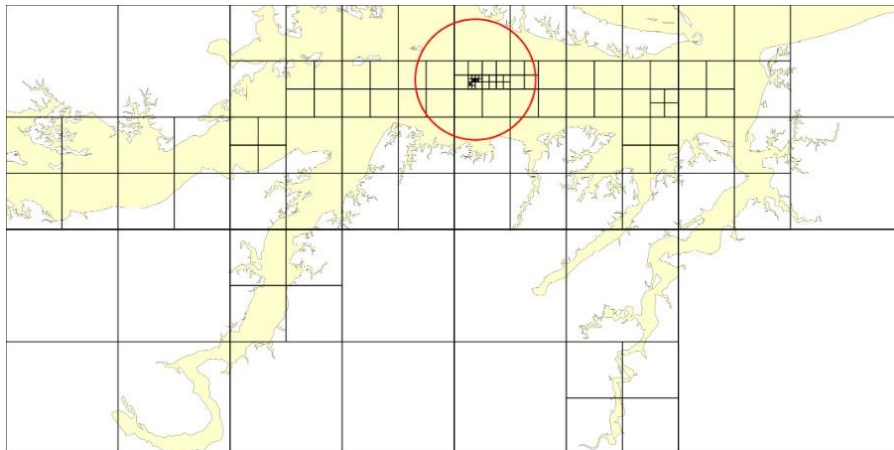


Figure 4. Polylabel sample

Algorithm stops when queue is exhausted and return the best center as pole of inaccessibility. Solution is written in JavaScript and available for public.

2.3 CAUV

Another algorithm [3] is proposed to solve the problem without dependence of existence of holes or convexity or non-convexity of polygon for CAUV project in Ohio. Method is focused on shape analysis such a compactness, weighted areas, region decompositions, and region skeletons. One of fundamentals of the algorithm is buffering. Buffering is a creation of zone around a point, line or polygon. Proposed algorithm is based on inside buffering a point that creates circle and main focus is finding biggest sub polygon or get skeleton by inside buffering until it becomes convex without holes. At the end label a position inside of polygon which is largest area of polygon. A region can be described by key factors such as compactness and rectangularity. These factors needed to find a label location for convex polygons and satisfaction of objectives of label placement.

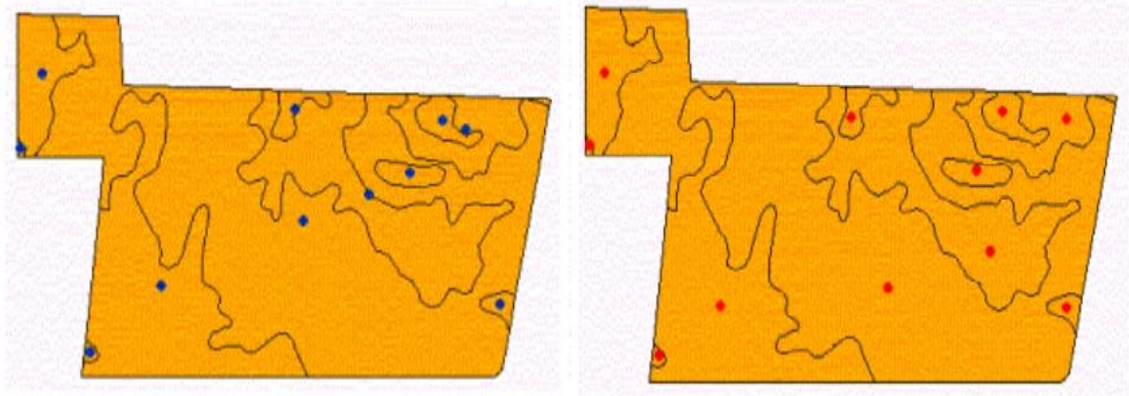


Figure 5. CAUV sample

Compactness is a ratio between the area and this square of border length of a shape. The higher values represent more winding shapes. Rectangularity is the ratio between the area of any polygon and the area of its bounding rectangle. When its value is equal to one, it represents perfect rectangular shape. Centroid (blue) points are represented by ArcView, while label (red) points are implemented in Avenue script using proposed algorithm in the figure 5. However, this algorithm does not work for circular or rectangular ring. In order to get solution another criterion should be added. Script of solution is not available for public.

2.4 MICGIS

In this study [4], it is assumed that MIC is located along the medial axis (MA) of polygon. The polygon's MA is approximated by Voronoi diagram. It is known that the MIC must be one of the

circles drawn around the MA and touching the polygon boundary in at least two points in the figure 6.

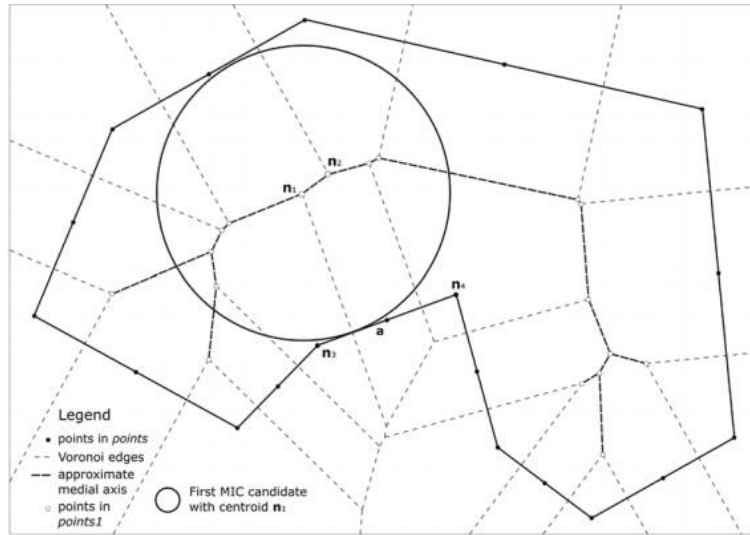


Figure 6. MICGIS sample

MICGIS algorithm consists of several steps:

1. Initially approximate MA is constructed and candidate points, that are located along Voronoi edge and covered by polygon, extracted as center of MIC.
2. The first candidate point for the center of MIC can be calculated maximizing the minimum distance from the point polygon's MA to all edges of polygon.
3. It is required to find the potential cores in the case of concave polygons, more than one core can approximate the center of MIC.
4. Candidate points are processed for approximation best MIC that must be tangent at least three edges. In the case of concave polygon special cases of Apollonius problem and other adjustments are applied.
5. BestMIC is selected by its full coverage by polygon and biggest one compared with the others.

Implementation of MICGIS has run in OpenJUMP software for regular, irregular and complex polygons.

2.5 Maximum inscribed circle

[5] MIC problem is solved using Distance Transform and Voronoi Diagram in Matlab by Tolga Birdal. Solution is available for public. The first approach is completely in precise and is not subpixel accurate. It only guarantees a global convergence.

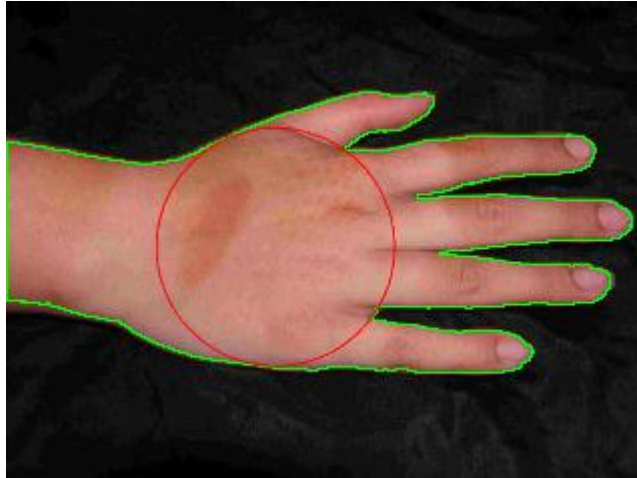


Figure 7. MIC using Voronoi diagram sample

The second approach, unlike the first one, is subpixel accurate in the figure 7. In general, algorithm has following steps:

- Construct Voronoi Diagram of polygon
- Find Voronoi nodes within polygon
- Maximize the distance of each Voronoi node to closest node on the polygon

3. Possible key parameters for grasping point of samples without micro-joint

3.1 Center of mass

First step on the detaching is identification grasping point of each work piece. Grasping point can be [6] center of mass (COM) since it is the point that is concentrated whole mass of object. Due to mathematical definition of COM, it is the unique point at the center of a distribution of mass in space that weighted position vectors relative to this point sum to zero.

$$\sum_{i=1}^n m_i \mathbf{r}_i = 0$$

$$\mathbf{R}_{\text{cm}} = \frac{1}{M} \int \mathbf{r} dm \text{ for continuous systems}$$

$$\mathbf{R}_{\text{cm}} = \frac{1}{M} \sum_{i=1}^n m_i \mathbf{r}_i \text{ for discrete systems}$$

Examples for discrete and continuous systems are a system of particles and single solids, respectively. Where each mass m_i that is located in space with position vector \mathbf{r}_i . If the origin we have choose for our vectors at the COM, then sum will be zero. Otherwise, vector sum \mathbf{R}_{cm} will shows us to the COM. Here M is the total mass. Vector and scalar coordinates of COM of 3D solids are defined as

$$\mathbf{R}_{\text{cm}} = \frac{1}{M} \iiint \mathbf{r} \, dm$$

$$x_{\text{cm}} = \frac{1}{M} \iiint x \, \rho dV$$

$$y_{\text{cm}} = \frac{1}{M} \iiint y \, \rho dV$$

$$z_{\text{cm}} = \frac{1}{M} \iiint z \, \rho dV$$

where $dm = \rho dV$, ρ is mass per unit volume.

Vector and scalar coordinates of COM of 2D solids are defined as

$$\mathbf{R}_{\text{cm}} = \frac{1}{M} \iint \mathbf{r} \, dm$$

$$X_{\text{cm}} = \frac{1}{M} \iint x \, \rho dA$$

$$y_{\text{cm}} = \frac{1}{M} \iint y \, \rho dA$$

where $dm = \rho dA$, ρ is mass per unit area.

Vector and scalar coordinates of COM of 1D solids are defined as

$$\mathbf{R}_{\text{cm}} = \frac{1}{M} \int \mathbf{r} \, dm$$

$$X_{\text{cm}} = \frac{1}{M} \int x \, \rho ds$$

where $dm = \rho ds$, ρ is mass per unit length.

For simplicity we can avoid vector arithmetic and COM of system of particles can be computed separately along each axis.

$$X_{\text{cm}} = \frac{\sum_{i=1}^n x_i m_i}{M}$$

$$X_{\text{cm}} = \frac{\sum_{i=1}^n x_i m_i}{M}$$

$$X_{\text{cm}} = \frac{\sum_{i=1}^n x_i m_i}{M}$$

$$y_{\text{cm}} = \frac{\sum_{i=1}^n y_i m_i}{M}$$

$$y_{\text{cm}} = \frac{\sum_{i=1}^n y_i m_i}{M}$$

$$z_{\text{cm}} = \frac{\sum_{i=1}^n z_i m_i}{M}$$

3D system

2D system

1D system

Total mass $M = \sum_{i=1}^n m_i$ for discrete systems
 $= \int dm$ for continuous systems

It could be the case, mentioned formulas above are not appropriate or difficult to apply. In this case it is required another approach.

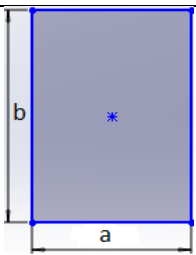
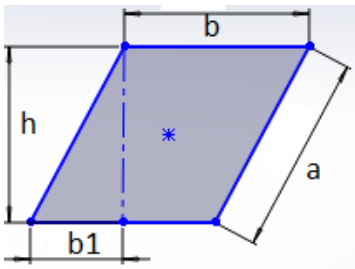
3.1.1 COM and Centroid

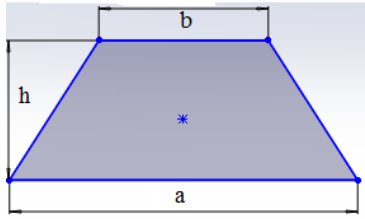
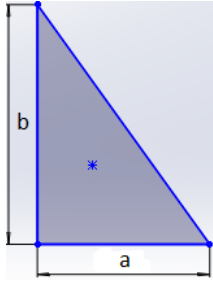
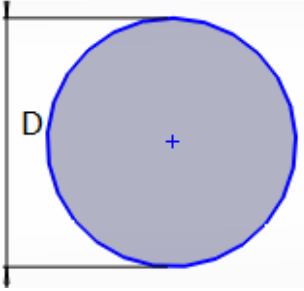
COM is located in the centroid if rigid bodies have uniform density. Mathematical point of view, uniform density refers constant density term in formulas. It allows to cancel constant density term in the expressions.

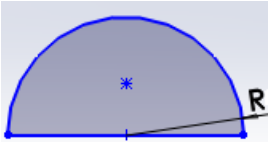
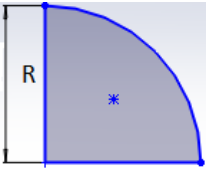
$$\mathbf{r}_{cm} = \frac{1}{M} \int \mathbf{r} dm = \frac{\int \mathbf{r} \rho dV}{\int \rho dV} = \frac{\rho \int \mathbf{r} dV}{\rho \int dV} = \frac{\int \mathbf{r} dV}{V}$$

$$\mathbf{r}_{cm} = \frac{\sum_{i=1}^n \mathbf{r}_i \rho V_i}{\rho V_T} = \frac{\rho \sum_{i=1}^n \mathbf{r}_i V_i}{\rho V_T} = \frac{\sum_{i=1}^n \mathbf{r}_i V_i}{V_T}$$

As a result, last expressions form geometric centroid for continuous and discrete systems. This behavior holds for 2D and 1D geometric shapes. Our interest focuses to study centroid more in 2D than 3D and 1D systems. In general COM and centroid are interchangeable this context.

Shape	Figure	\bar{X}	\bar{Y}	Area
Rectangle		$\frac{a}{2}$	$\frac{b}{2}$	ab
Parallelogram		$\frac{b + b1}{2}$	$\frac{h}{2}$	hb

Isosceles trapezoid		$\frac{a}{2}$	$\frac{h(2a + b)}{3(a + b)}$	$\frac{h(a + b)}{2}$
Right angle triangle		$\frac{a}{3}$	$\frac{b}{3}$	$\frac{ab}{2}$
Circle		X=0 or X coordinate of origin	Y=0 or Y coordinate of origin	$\frac{\pi D^2}{4}$

Semicircle		X=0	$\frac{4R}{3\pi}$	$\frac{\pi R^2}{2}$
Quarter circle		$\frac{4R}{3\pi}$	$\frac{4R}{3\pi}$	$\frac{\pi R^2}{4}$

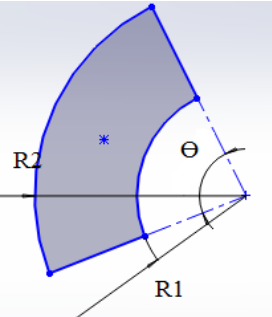
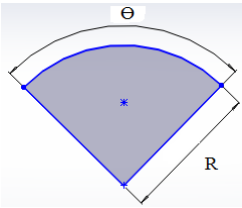
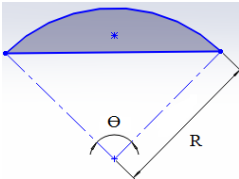
Ring segment		$\frac{\iint r \cos \theta \, r \, dr \, d\theta}{\iint r \, dr \, d\theta}$	$\frac{\iint r \sin \theta \, r \, dr \, d\theta}{\iint r \, dr \, d\theta}$	$\iint r \, dr \, d\theta$
Circular sector		X=0	$\frac{4R \sin \frac{\theta}{2}}{3\theta}$	$\frac{\theta}{2} R^2$
Circular segment		X=0	$\frac{4R \sin^3 \frac{\theta}{2}}{3(\theta - \sin \theta)}$	$\frac{R^2}{2} (\theta - \sin \theta)$

Table 1. Centroids of common shapes

3.1.2 COM of complex shapes

Plumb line method is another experimental determination that can be useful for objects that have complex planar shape with uniform density.

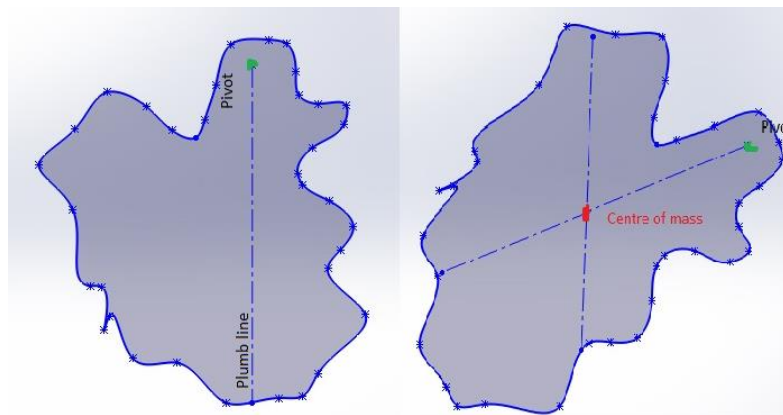


Figure 8. Plumb line method sample

Irregular object is suspended on a pin, as soon as under gravity reaches a stable point and plumb line is dropped to mark the object. The pin is moved to another location and procedure is repeated. The center of mass lies intersection of two lines in the figure 8. This method is easy to apply depending the size of shape, but it is not good enough in terms of accurate numerical coordinates.

It can be considered for calculation purposes; mass of rigid bodies can be treated as a point mass since it can be assumed mass of solid is concentrated in a tiny object at the center of mass.

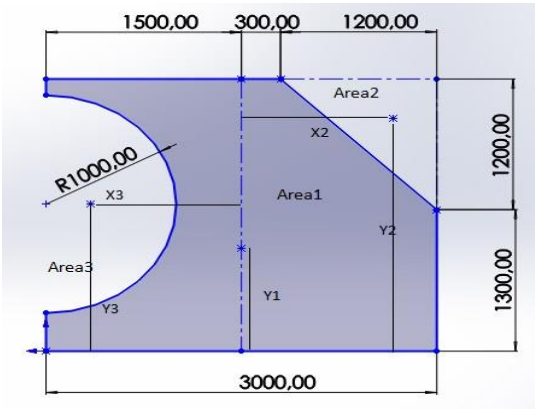


Figure 9. Area decomposition sample

Composite shapes can be also formed by collection of simple shapes such as rectangular, circle or semi-circle, and triangular, each with uniform mass. Voids within objects can be represented as shapes with negative mass. The complex shape in the figure 9 consists of a rectangle, right triangle and semicircle, and they are divided Area1, Area2, and Area3, respectively. Note that area of rectangle is multiplication of full length and width. (Area1=3000x2500). It is required to subtract areas of semicircle and right triangle, since they are void, from the area of rectangle. The dashed line is y axis of the shape, while x axis is laid at the bottom length of shape. It can be seen also centroid coordinates of each area. Units are given in millimeters, for simplicity it is converted to meter. Coordinates x2 and x3 are found as following: $x_2=30+(2b/3)$, $x_3=15-(4r/3\pi)$ where b is the side of triangle, while r is radius of semicircle.

Area Name	Area	x	y	Ax	Ay
Area1	750	0	12.5	0	9375
Area2	-72	11	21	-792	-1512
Area 3	-157	-10.81	13.5	1697.17	-2119.5
Total	521			905.17	5743.5

$$x_{cm} = \frac{\sum Ax}{\sum A} : x_{cm} = 1.7373 \text{ m} = 173.73\text{mm}$$

$$y_{cm} = \frac{\sum Ay}{\sum A} : y_{cm} = 11.024\text{m} = 1102.4\text{mm}$$

3.1.3 The center of gravity

The center of gravity is the point through which the force of gravity acts on an object or system. Gravitational field is assumed uniform in most mechanics problems. The center of gravity is exactly the same point as the center of mass. These terms are used interchangeably. If gravitational field is non-uniform, the center of mass and center of gravity will be at two different locations.

3.2 Inertia Moments

The second important parameter for grasping point could be one point along principal inertia axes of object. It is seen differences between Area Moment of Inertia and Mass Moment of Inertia.

3.2.1 Area moments of inertia

Area moments of inertia is an important parameter studies bending resistance behavior of a cross section of object under a certain load. It is also called as second moment of area or second moment of inertia. Unit of area moments of inertia is meter in power four (m^4) according to International Systems of Unit.

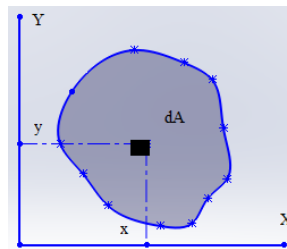


Figure 10. Area moments of Inertia sample

The second moment of area for arbitrary shape with respect to arbitrary axis AA' is defined as

$$I_{AA'} = \iint r^2 dA$$

Where dA is the differential area of arbitrary shape, r is the distance from the axis AA' to dA in the figure 10. The second of moment of inertia can be computed in Cartesian coordinate system with respect to x and y axes as following:

$$I_x = \iint y^2 dx dy,$$

$$I_y = \iint x^2 dx dy.$$

Product moment of area has an expression as

$$I_{xy} = \iint yx dx dy.$$

According to Perpendicular Axis theorem, the second moment of area along z axis can be formulated as

$$I_z = I_x + I_y.$$

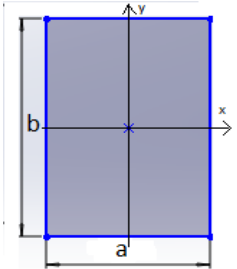
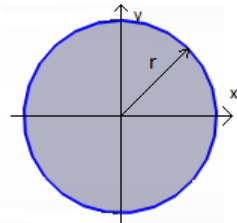
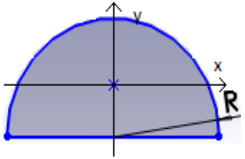
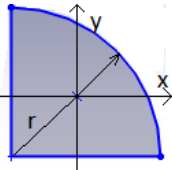
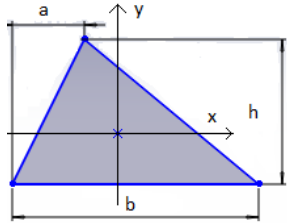
Shape	Figure	Moment of Area
Rectangle		$I_x = \frac{b^3 a}{12}$ $I_y = \frac{a^3 b}{12}$ $I_z = \frac{1}{12} ba(b^2 + a^2)$
Circle		$I_x = \frac{\pi}{4} r^4$ $I_y = \frac{\pi}{4} r^4$ $I_z = \frac{\pi}{2} r^4$
Semicircle		$I_x = \left(\frac{\pi}{8} - \frac{8}{9\pi}\right) r^4$ $I_y = \frac{\pi}{8} r^4$ $I_z = \left(\frac{\pi}{4} - \frac{8}{9\pi}\right) r^4$
Quarter circle		$I_x = \left(\frac{\pi}{16} - \frac{4}{9\pi}\right) r^4$ $I_y = \left(\frac{\pi}{16} - \frac{4}{9\pi}\right) r^4$ $I_z = \left(\frac{\pi}{8} - \frac{8}{9\pi}\right) r^4$
Triangle		$I_x = \frac{bh^3}{36}$ $I_y = \frac{b^3 h - b^2 ha + bha^2}{36}$ $I_z = \frac{b^3 h - b^2 ha + bha^2 + bh^3}{36}$

Table 2. Origin of reference frame is located at the centroid of shape

Thanks to Parallel Axis theorem, the second moment of area can be computed any axis parallel to centroidal axis

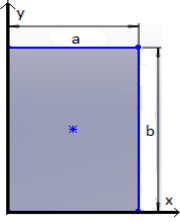
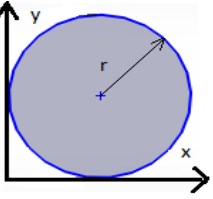
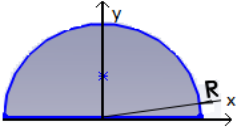
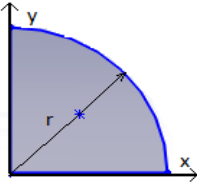
$$I_{x,O} = I_{x,C} + Ad_y^2,$$

$$I_{y,O} = I_{y,C} + Ad_x^2$$

where $I_{x,O}$, $I_{y,O}$ are the second moment of area parallel centroidal axes, $I_{x,C}$, $I_{y,C}$ are the second moment of area about centroidal axes, A is area of shape and d_x , d_y are perpendicular distances between parallel axes and x, y centroidal ones, respectively. For product moment of area, expression parallel axis theorem becomes

$$I_{xy,O} = I_{xy,C} + Ad_xd_y$$

where $I_{xy,O}$ is product moment of area that are parallel to centroidal x, y axes, $I_{xy,C}$ is product moment of area about x, y centroidal ones.

Shape	Figure	Moment of Area
Rectangle		$I_x = \frac{b^3 a}{3}$ $I_y = \frac{a^3 b}{3}$ $I_z = \frac{1}{3} ba(b^2 + a^2)$
Circle		$I_x = \frac{5\pi}{4} r^4$ $I_y = \frac{5\pi}{4} r^4$ $I_z = \frac{5\pi}{2} r^4$
Semicircle		$I_x = \frac{\pi}{8} r^4$ $I_y = \frac{\pi}{8} r^4$ $I_z = \frac{\pi}{4} r^4$
Quarter circle		$I_x = \frac{\pi}{16} r^4$ $I_y = \frac{\pi}{16} r^4$ $I_z = \frac{\pi}{8} r^4$

Triangle		$I_x = \frac{bh^3}{12}$ $I_y = \frac{b^3h - b^2ha + bha^2}{12}$ $I_z = \frac{b^3h - b^2ha + bha^2 + bh^3}{12}$
----------	--	--

Table 3. Origin of reference frame is located different from centroid

C type shape is given with its dimensions. It is required to find the second moment of area tensor.

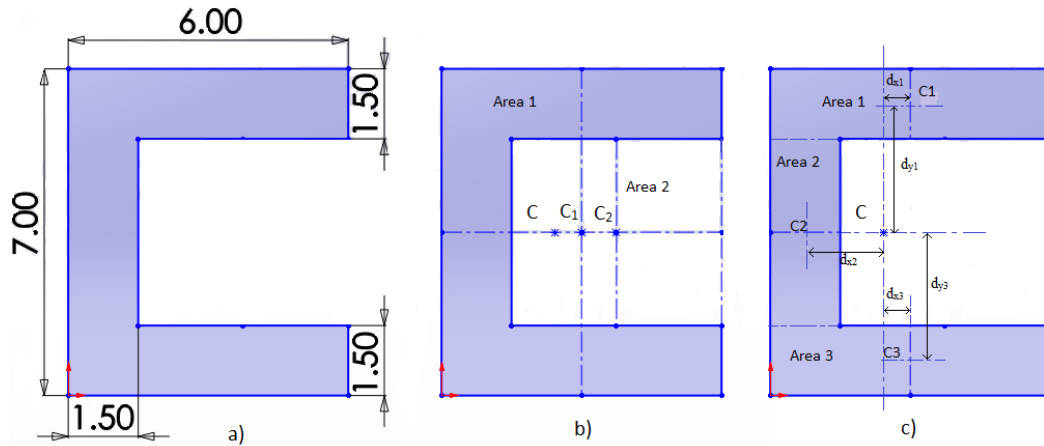


Figure 11. Second Moment of Area of Composite Shape

On the first step, centroid can be computed by area decomposition approach. Here C type cross section is divided into two rectangles, Area 1 and Area 2 in the figure 11 b. Area 2 is empty or hollow rectangle while Area 1 is full one. Centroids of C cross section, Area 1 and Area 2 are shown by C, C₁ and C₂, respectively.

Area Name	Area	x	y	Ax
Area1	42	3	3.5	126
Area2	-18	3.75	3.5	-67.5
Total	24			58.5

$$x_{cm} = \frac{\sum Ax}{\sum A} = 2.44$$

$$y_{cm} = 3.5$$

The same area decomposition works for identification of second moment area. It is also possible to divide C shape into three rectangles in the figure 11 c and to apply Parallel Axis theorem in both cases. Computation of both cases are given below.

a) Area decomposition into two

i	A _i (area)	\bar{x}	\bar{y}	I _{x,i} w.r.t own centroid	I _{y,i} w.r.t own centroid	d _{xi}	d _{yi}	A _i *d _{xi} ²	A _i *d _{yi} ²	I _{xxi} = I _{x,i} +A _i *d _{yi} ²	I _{yyi} = I _{y,i} +A _i *d _{xi} ²
1	42	3	3.5	171.5	126	0.56	0	13.17	0	171.5	139.17
2	18	3.75	3.5	24	30.375	1.31	0	30.89	0	24	61.265

$$I_{x,C} = I_{xx1} - I_{xx2} = 147.5 \text{ cm}^4$$

$$I_{y,C} = I_{yy1} - I_{yy2} = 77.905 \text{ cm}^4$$

$$I_{z,C} = I_{x,C} + I_{y,C} = 225.405 \text{ cm}^4$$

a) Area decomposition into three

i	A _i (area)	\bar{x}	\bar{y}	I _{x,i} w.r.t own centroid	I _{y,i} w.r.t own centroid	d _{xi}	d _{yi}	A _i *d _{xi} ²	A _i *d _{yi} ²	I _{xxi} = I _{x,i} +A _i *d _{yi} ²	I _{yyi} = I _{y,i} +A _i *d _{xi} ²
1	9	3	6.25	1.69	27	0.56	2.75	2.82	68.06	69.75	29.82
2	6	0.75	3.5	8	1.125	1.69	0	17.14	0	8	18.265
3	9	3	0.75	1.69	27	0.56	2.75	2.82	68.06	69.75	29.82

$$I_{x,C} = \sum_i^n I_{xx,i} = 147.5 \text{ cm}^4$$

$$I_{y,C} = \sum_i^n I_{yy,i} = 77.905 \text{ cm}^4$$

$$I_{z,C} = I_{x,C} + I_{y,C} = 225.405 \text{ cm}^4$$

$I_{x,C}$, $I_{y,C}$, $I_{z,C}$ are the second moment of area of C shape about its centroid. Products of moment of area I_{xy} , I_{yx} can be positive, negative and zero. Either x or y axis of centroid is axis of symmetry, I_{xy} , I_{yx} becomes zero.

$$\mathbf{I} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} = \begin{bmatrix} 147.5 & 0 & 0 \\ 0 & 77.905 & 0 \\ 0 & 0 & 225.405 \end{bmatrix} \text{ cm}^4$$

3.2.2 Mass moment of inertia

Unfortunately, second moment of area is also called moment of inertia in some context and notation of both concepts are the same. Mass moment of inertia is a capability of resistance of object for rotational motion. Unit of dimension of moment of inertia is kilogram meter square [kg m²]. It is better to distinguish two quantities looking at their unit of dimension.

$$I = \int r^2 dm,$$

$$\mathbf{I} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

Where I is a mass moment of inertia, \mathbf{I} is an inertia tensor that represents how mass is distributed in a rigid body and it has scalar and vector behavior. I_{xx} , I_{yy} , I_{zz} are mass moment of inertia, and given by

$$I_{xx} = \int (y^2 + z^2) dm, \quad I_{yy} = \int (x^2 + z^2) dm \quad \text{and} \quad I_{zz} = \int (y^2 + x^2) dm.$$

The rest elements I_{xy} , I_{xz} , I_{yx} , I_{yz} , I_{zx} and I_{zy} are called products of inertia, and given by

$$I_{xy} = I_{yx} = \int xy dm, \quad I_{xz} = I_{zx} = \int zx dm \quad \text{and} \quad I_{yz} = I_{zy} = \int zy dm.$$

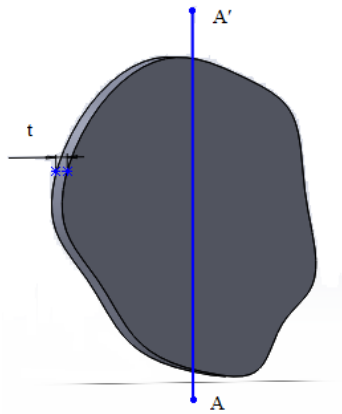


Figure 12. Mass Moment of Inertia sample

In case of a thin plate has a uniform thickness t in the figure 12, mass moment of inertia about an axis AA' can be written

$$I_{AA', \text{mass}} = \int r^2 dm,$$

$$I_{AA', \text{mass}} = \rho t \int r^2 dA,$$

$$I_{AA', \text{mass}} = \rho t I_{AA', \text{Area}}$$

where $\rho t dA = dm$, ρ is constant density per volume. It can be applied for C type cross section example. It can be assumed $t = 2.5$ cm and ρ is 1 g/cm^3 . Mass moment of inertia of C type shape becomes

$$\mathbf{I} = \begin{bmatrix} 368.75 & 0 & 0 \\ 0 & 194.76 & 0 \\ 0 & 0 & 563.51 \end{bmatrix} \text{g} \cdot \text{cm}^2$$

The physical meaning of product of inertia is as following: when torque is applied along x axis and products of inertia (I_{xy} , I_{xz}) are non-zero, applied torque has components along y and z axis, respectively.

$$\mathbf{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

If the tensor of inertia is diagonal, these axes represent the principal axes of inertia. Minimum torque is required for rotation along principal axes. That's why it is desired to get a candidate grasping point along principal axes.

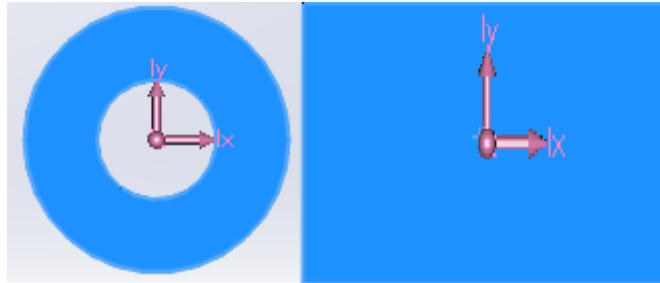


Figure 13. Principal axes in common shapes

For symmetric bodies, it can be visible which axis is principal. The symmetry axis of the shape, is also considered a principal axis. For a ring and a rectangle, axes x and y are both symmetry axes in the figure 13. Even though bodies without symmetries, three orthogonal principle axes of inertia exist in 3D systems.

3.2.3 The Search for Principal Axes as Eigenvalue Problem

It can be considered general non-symmetrical body with unknown principal axis in the figure 14. For rotation about the principle axis, angular momentum and angular velocity are the same direction.

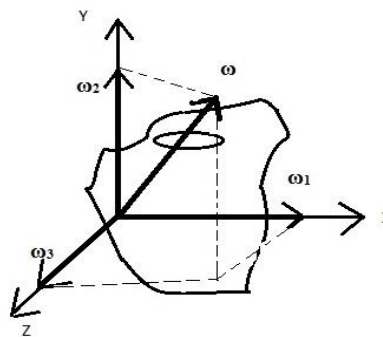


Figure 14. Principal axes in 3D

It can be sought coordinate axes x, y and z, about which is a rotation ω_x , ω_y and ω_z which is aligned with this coordinate direction. Coordinate axes will be parallel to angular momentum vector and formed by equation

$$\mathbf{H} = \begin{bmatrix} I\omega_x \\ I\omega_y \\ I\omega_z \end{bmatrix} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

It may be expressed general form of angular momentum vector and general form of inertia tensor along x, y and z axis.

$$\mathbf{H} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

To obtain special direction of ω that is aligned with a principal axis, we equate these two expression.

$$\mathbf{H} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

$$\mathbf{H} = \begin{bmatrix} (I_{xx} - I) & I_{xy} & I_{xz} \\ I_{yx} & (I_{yy} - I) & I_{yz} \\ I_{zx} & I_{zy} & (I_{zz} - I) \end{bmatrix} \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

The solution of this structure represents the principal axes and their magnitudes. Three eigenvalues give the direction of three principal axes and three eigenvectors give the moments of inertia with respect to each of these axis.

$$\mathbf{H} = I_{xx}\omega_x\mathbf{i} + I_{yy}\omega_y\mathbf{j} + I_{zz}\omega_z\mathbf{k}$$

3.3 Medial Axis and Its Computation Methods

Another candidate point for grasping can be the one that is located on the medial axis. Medial Axis or Straight Skeleton [7] of a shape is a thin or core version of that shape that is equidistant to its boundaries. Medial Axis (MA) is a method of representing a polygon by a topological skeleton. A straight skeleton is a part of MA, in the case of convex polygon both are the same. The difference of between them is existence of parabolic curves as well as straight line segments in MA in non-convex polygons. The straight skeleton of polygon can be done wavefront propagation process. Each edge of polygon sends out wavefront which moves inward at a constant speed and is parallel to the edge. Edges collapse to zero length and wavefront may be split into multiple parts. The straight line segments are called the arcs of straight skeleton. The loci of topological changes of wavefront are called nodes. Each edge of polygon belongs to face in figure 15. Polygon is described in bold. Wavefronts moves inwards in a self-parallel manner from the edges of polygon. Five wavefronts

spread at equally spaced points in time are shown in gray. Straight skeleton is drawn in blue and straight segments are called the arcs of straight skeleton. Common end points of arcs are called node of straight skeleton. It is shaded face $f(e)$ of one edge e in light gray. Straight skeleton can be applied to solve problems in the field of offset curves and NC-machining, building roofs and generating terrains, shape reconstruction and contour interpolation, polygon decomposition and so on.

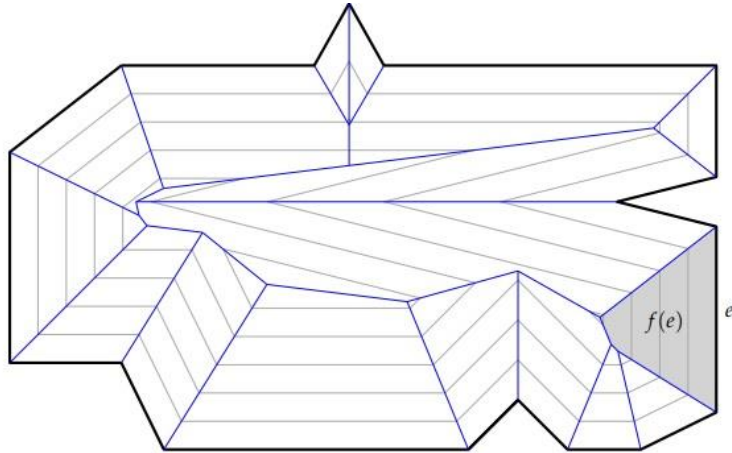


Figure 15. Straight skeleton of polygon

In general there are four approaches for MA computation. These approaches are different in terms of underlying representation and computation of medial axis.

3.3.1 Thinning algorithms

Thinning algorithms [8] perform erosion operations to approximate the MA. These methods play a significant role in the image processing and pattern recognition areas.

3.3.2 Distance Field Computations

Most of algorithms compute MA using distance fields. Spatial subdivision [9] is used for representing the MA, for further subdivision of a cell can be determined on the nearest neighbour queries. Subdivision occurs cell vertices in different Voronoi regions and is larger a certain threshold.

Another MA approximating algorithm [10] is based on a differential equation simulating inward progress from boundary of object. Vector field is computed, at every point \mathbf{p} , is equal to the vector from the nearest point on the surface. It is considered that a point is on the medial axis if mean flux of vector field and positive.

3.3.3 Algebraic Methods

Spatial subdivision resolves the connectivity of curves, curves and surfaces are represented symbolically. The surface pass into the cell subdivision is determined by algebraic tests and subdivision occurs until minimum cell size is reached.

It can be seen most algorithms MA symbolic representation is based on a tracing approach. It is started from a junction point on the MA, a seam emanating from the junction is followed. Using this approach, MA [11] computation is represented in terms of planar region bounded by piecewise C^2 curves.

Application of all methods in this family is based on polyhedra with a few hundred faces. It is not clear to apply it for hundreds of thousands of faces.

3.3.4 Surface sampling approaches

Voronoi vertices dual to tetrahedra which is a subset of Voronoi diagram, approximates the MA. [12] suggests a simplified surface model of MA. These algorithms to models consisting of tens of thousands of points. In the worst case running time can be $O(n^2)$, where n is number of sampling points.

3.3.5 Medial Axis construction through Voronoi diagram

Voronoi diagram [17] is used in many fields for construction algorithms in geometrical computation. Voronoi diagram is a big and complex topic and MA is the subset of it. The most important property of Voronoi diagram is Voronoi vertices approximation the MA of polygon. Voronoi diagram and MA is the same in case of convex polygons.

4. Applications of key parameters and methods

4.1 Center of mass computation

eDrawing of metal sheet with all elements in .dwg file is provided. It shows square metal sheet with its side length 500 inches (12700 mm). In real case, metal sheet has a size with 500mm x 500mm. This problem could be generated with its scale while saving the file. So the first step is modifying the scale in order to get appropriate results. The ratio between 12700 mm and 500 mm forms 25.4 ($12700 \div 500 = 25.4$). This ratio can be represented in terms of scale in 10:254 form. It leads to get correct sizes. There are 7 different type of shapes and they are 33 elements in total in the figure 16.

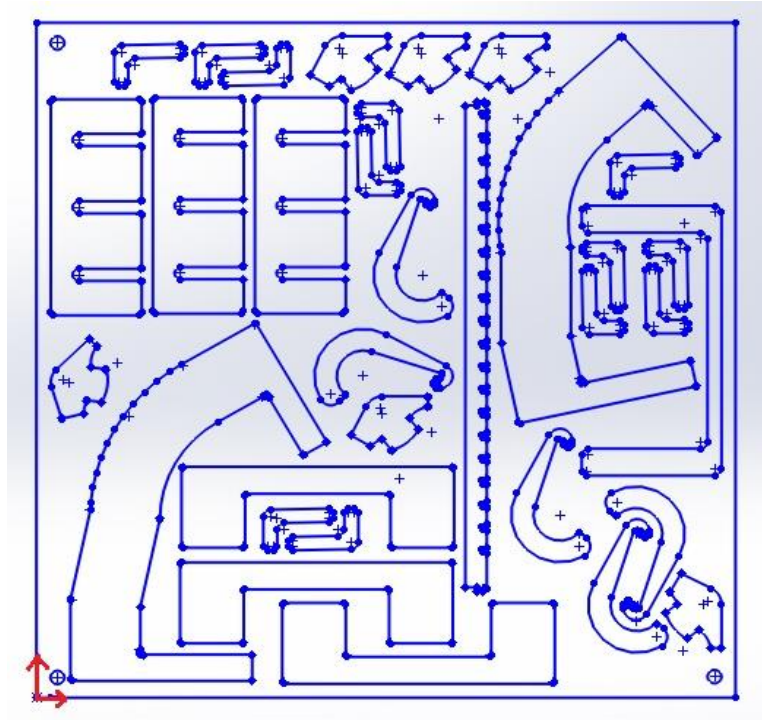


Figure 16. Work piece sketch in CAD

Origin is located bottom left corner of the metal sheet by default. Initial task is to explore the mathematical methods that identifies the center of mass of each element of the metal sheet and compare the results obtained in Solidworks. This file is converted into Solidworks 2D sketch in order to take advantage by measuring the size and using some additional center lines for computational reasons. The level difficulty of computation of COM of each element depends on directly its complexity. If element composed of one or set of figures such that whole or piece circle, rectangle, triangle, trapezoid and the other commonly used ones, centroid can be found easily. In case of element has irregular shape, it is not easy to describe its entire area mathematically.

4.1.1 Manual computation of center of mass

There are different methods identifying COM, but it is hard to find a universal method which is suitable all or most of irregular figures since mathematical description is unknown. Fortunately, irregular shapes of metal sheet consist of a set of commonly used and mathematically known figures. This gives an advantage to use 2D COM equations with geometrical decomposition approach. Each element is computed with this method. Units are measured in millimeters.

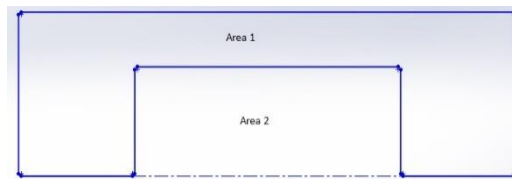


Figure 17. Sample 1

The first shape is “C” type in the figure 17. There are 3 such kind of shapes. Shape decomposed 2 rectangle areas. Centroid formula of rectangle is used. Fillet radius of shapes are neglected because of symmetry and negligible size. It is enough to compute only y_{cm} because of its symmetry.

a)

Area Name	Area	x	y	Ax	Ay
Area1	11700	273.26	40.24	3197142	470808
Area2	-4160	273.26	50.24	-1136761.6	-208998.4
Total	7540			2060380.4	198809.6

$$X_{cm} = \frac{\sum Ax}{\sum A} = 273.26; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 34.66$$

b)

Area Name	Area	x	y	Ay
Area1	11700	200.26	70.24	821808
Area2	-4160	200.26	60.24	-250598.4
Total	7540			571209.6

$$X_{cm} = 200.26; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 75.76$$

c)

Area Name	Area	x	y	Ay
Area1	11700	201.26	141.07	1650519
Area2	-4160	201.26	131.07	-545251.2
Total	7540			1105267.8

$$X_{cm} = 201.26; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 146.59$$



Figure 18. Sample 2

d) Another C type shape, there is a difference on its size in the figure 18.

Area Name	Area	x	Ax
Area1	20000	440	3197142
Area2	-14400	435	-1136761.6
Total	5600		2060380.4

$$X_{cm} = \frac{\sum Ax}{\sum A} = 452.86; \quad y_{cm} = 264.48$$

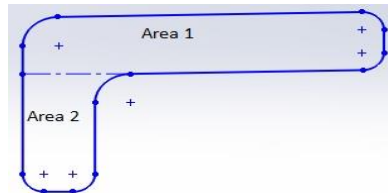


Figure 19. Sample 3

In this case “L” shape is divided into 2 rectangular, Area 1 and Area 2 as indicated in the figure 19.

a)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	205.26	114.17	102609.47	57073.58
Area2	209.5	225.25	129.61	47189.88	27153.295
Total	709.4			149799.35	84226.875

$$X_{cm} = \frac{\sum Ax}{\sum A} = 211.16; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 118.73$$

b)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	187.255	134.46	93608.77	67216.55
Area2	209.5	167.26	119.635	35040.97	25063.53
Total	709.4			128649.74	92280.08

$$X_{cm} = \frac{\sum Ax}{\sum A} = 181.35; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 130.08$$

c)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	80.675	478.57	40329.43	239237.14
Area2	209.5	60.68	463.745	12712.46	97154.58
Total	709.4			53041.89	336391.72

$$X_{cm} = \frac{\sum Ax}{\sum A} = 74.77; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 473.19$$

d)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	138.665	480.51	69318.63	240206.95
Area2	209.5	118.67	463.685	24861.365	97142
Total	709.4			94179.995	337348.95

$$X_{cm} = \frac{\sum Ax}{\sum A} = 132.76; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 475.54$$

e)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	435.605	396.97	217758.94	198445.303
Area2	209.5	415.61	382.085	87070.295	80046.808
Total	709.4			304829.235	278498.11

$$X_{cm} = \frac{\sum Ax}{\sum A} = 429.7; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 392.57$$

f)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	156.665	458.21	78316.83	229059.179
Area2	209.5	176.66	473.685	37010.27	99237
Total	709.4			115327.1	328296.179

$$X_{cm} = \frac{\sum Ax}{\sum A} = 162.57; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 462.78$$

g)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	255.21	415.145	127579.479	207530.99
Area2	209.5	239.735	435.14	50224.48	91161.83
Total	709.4			177803.959	298692.82

$$X_{cm} = \frac{\sum Ax}{\sum A} = 250.64; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 421.05$$

h)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	235.11	397.145	117531.489	198532.79
Area2	209.5	250.385	377.15	52455.66	79012.925
Total	709.4			169987.149	277545.715

$$X_{cm} = \frac{\sum Ax}{\sum A} = 239.62; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 391.24$$

i)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	415.92	312.135	207918.408	156036.29
Area2	209.5	400.455	332.13	83895.32	69581.235
Total	709.4			291813.728	225617.525

$$X_{cm} = \frac{\sum Ax}{\sum A} = 411.35; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 318.04$$

j)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	395.82	294.135	197870.418	147038.09
Area2	209.5	411.095	274.14	86124.4	57432.33
Total	709.4			283994.818	204470.42

$$X_{cm} = \frac{\sum Ax}{\sum A} = 400.33; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 288.23$$

k)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	461.16	312.855	230533.884	156396.2
Area2	209.5	445.685	332.85	93371	69732.08
Total	709.4			323904.884	226128.28

$$X_{cm} = \frac{\sum Ax}{\sum A} = 456.59; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 318.76$$

l)

Area Name	Area	x	y	Ax	Ay
Area1	499.9	441.06	294.855	220485.894	147398.01
Area2	209.5	456.335	274.86	95602.18	57583.17
Total	709.4			316088.074	20981.18

$$X_{cm} = \frac{\sum Ax}{\sum A} = 445.57; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 288.95$$

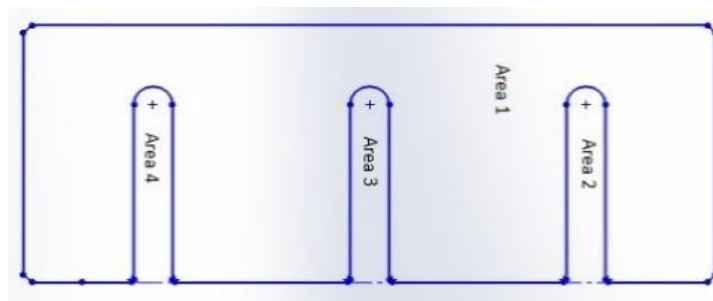


Figure 20. Sample 4

The sample 4 shape divided into 4 areas in the figure 20. Area 1 is rectangular with full length and width. The other areas (Area 2, Area 3, Area 4) are the same voids and combination of semicircle and small rectangular.

a)

Area Name	Area	x	Ax
Area1	10400	42.76	444704
Area2	-481.79	53.48	-25766.13
Area3	-481.79	53.48	-25766.13
Area4	-481.79	53.48	-25766.13
Total	8954.63		367405.61

$$X_{cm} = \frac{\sum Ax}{\sum A} = 41.03; \quad y_{cm} = 363.14$$

b)

Area Name	Area	x	Ax
Area1	10400	115.76	1203904
Area2	-481.79	126.48	-60936.8
Area3	-481.79	126.48	-60936.8
Area4	-481.79	126.48	-60936.8
Total	8954.63		1021093.6

$$X_{cm} = \frac{\sum Ax}{\sum A} = 114.03; \quad y_{cm} = 364.48$$

c)

Area Name	Area	x	Ax
Area1	10400	188.76	1963104
Area2	-481.79	199.48	-96107.47
Area3	-481.79	199.48	-96107.47
Area4	-481.79	199.48	-96107.47
Total	8954.63		209347.7

$$X_{cm} = \frac{\sum Ax}{\sum A} = 17.03; \quad y_{cm} = 364.48$$

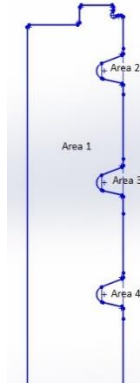


Figure 21. Sample 5

Only a piece of upper part of the sample 5 shape is described in this figure 21. In general, shape is considered rectangular neglecting upper and lower edges since it is symmetrical. Void area consists of circular segment and isosceles trapezoid. There are the same (N =20) void areas which consist of circular sector and isosceles trapezoid along y axis.

Area Name	Area	N	AN	x	ANx
Rect. Area	5562.84	1	5562.84	314.435	1749151.6
Void Area	-12.814	20	-256.28	319.4	-81855.88
Total			5306.56		1667295.77

$$X_{cm} = \frac{\sum Ax}{\sum A} = 314.19; \quad y_{cm} = 260.24$$

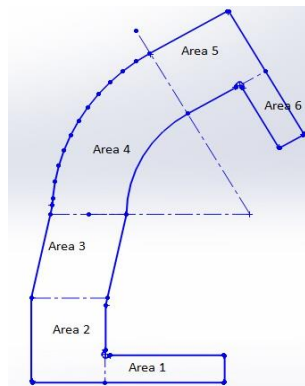


Figure 22. Sample 6

Sample 6 shape consists of 4 rectangles (Area 1, Area 2, Area 5 and Area 6), one parallelogram (Area 3) and one piece of ring (Area 4) in the figure 22.

a)

Area Name	Area	x	y	Ax	Ay
Area1	1560	114.32	22.17	178339.2	34585.2
Area2	2753	49.32	42.585	135777.95	117236.505
Area3	3379.032	57	101.56	192604.82	346553.5
Area4	5640.87	80.41	184.91	453582.36	1043053.27
Area5	3001.3	142.85	240.955	428735.56	723178
Area6	1014	186.375	211.435	188984.25	214395.09
Total	17348.201			1578024.08	2479001.59

$$X_{cm} = \frac{\sum Ax}{\sum A} = 90.96; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 142.9$$

b)

Area Name	Area	x	y	Ax	Ay
Area1	1560	430.99	231.765	672344.4	361553.4
Area2	2753	363.11	237.985	999641.81	655172.705
Area3	3379.032	357.31	301.445	1207361.92	1018592.301
Area4	5640.87	363	383.05	2047635.81	2160735.25
Area5	3001.3	412.81	451.745	1238966.24	1355821.82
Area6	1014	462.45	427.53	468919.23	433515.42
Total	17375.201			6589869.41	5985390.9

$$X_{cm} = \frac{\sum Ax}{\sum A} = 379.86; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 345.02$$

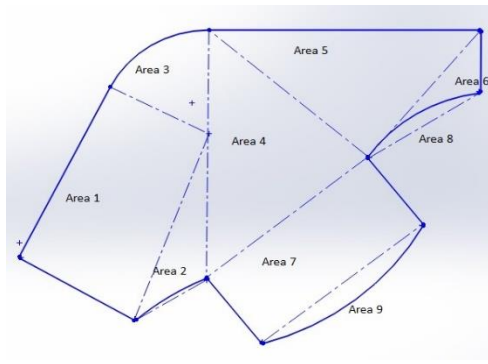


Figure 23. Sample 7

The sample 7 shape is complex in the figure 23. Fortunately, it consists of straight line and circular sector. As a result, it can be divided into simple areas such as a trapezoid (Area 1), a rectangle (Area 7), four triangles (Area 2, Area 4, Area 5 and Area 6), a circular sector (Area 3) and two circular segments (Area 8 and Area 9) which can be described by mathematical formula. Note that Area 6 is triangle with void circular segment.

a)

Area Name	Area	x	y	Ax	Ay
Area1	355.78	264.17	468.69	93986.4	166750.53
Area2	77.89	271.85	462.76	21174.4	36044.38
Area3	96.06	270.49	483.92	25983.27	46485.36
Area4	299.73	281.1	474.07	84254.1	142093
Area5	263.99	292.03	484.59	77093	127926.91
Area6	55.5	302.86	481.92	16823.87	26770.66

Area7	264.158	287.5	461.89	75961.85	122012.86
Area8	-18.2	300.21	478.47	-5463.82	-8708.15
Area9	30.3	291.58	456.92	11858.56	18582.94
Total	1435.63			401671.63	677958.49

$$X_{cm} = \frac{\sum Ax}{\sum A} = 279.79; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 472.24$$

Since it is known centroids of this shape it can be copied to others and it is enough to know coordinates.

b) $X_{cm} = 337.23$, $y_{cm} = 472.24$

c) $X_{cm} = 223.52$, $y_{cm} = 472.24$

d) $X_{cm} = 252.51$, $y_{cm} = 205.21$

e) $X_{cm} = 30.17$, $y_{cm} = 233.86$

f) $X_{cm} = 472.24$, $y_{cm} = 64.34$

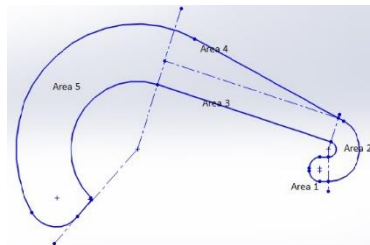


Figure 24. Sample 8

Sample 8 consists of 5 areas in the figure 24. They are two pieces of ring (Area 5 and Area 2), rectangle (Area 3), triangle (Area 4), rectangle with semi-circle (Area 1). Note that fillet of Area 5 is neglected.

a)

Area Name	Area	x	y	Ax	Ay
Area1	61.37	380.09	187.24	23326.12	11490.92
Area2	107.61	375.18	194.65	40373.12	20946.29
Area3	363.97	361.09	167.105	131426.47	60821.46
Area4	252.42	352.01	160.78	88854.36	40584.09
Area5	1175.62	364.5	118.01	428513.49	138739.92
Total	1960.37			712493.56	272582.68

$$X_{cm} = \frac{\sum Ax}{\sum A} = 363.33; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 139$$

b)

Area Name	Area	x	y	Ax	Ay
Area1	61.37	423.15	69.55	25968.72	4268.28
Area2	107.61	428.07	62.14	46064.61	6686.89
Area3	363.97	442.15	89.685	160929.34	32642.65
Area4	252.42	451.32	96.13	113922.19	24265.13
Area5	1175.62	438.44	138.95	515438.83	163352.4
Total	1960.37			862323.69	235183.63

$$X_{cm} = \frac{\sum Ax}{\sum A} = 439.88; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 119.97$$

c)

Area Name	Area	x	y	Ax	Ay
Area1	61.37	435.18	120.23	26707	7378.52
Area2	107.61	430.27	127.64	46301.35	13735.34
Area3	363.97	416.18	101	151477.03	36435.22
Area4	252.42	407.46	94.24	102851.05	23788.06
Area5	1175.62	419.59	51	493278.4	59956.62
Total	1960.37			684614.48	141293.76

$$X_{cm} = \frac{\sum Ax}{\sum A} = 418.6; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 72.08$$

d)

Area Name	Area	x	y	Ax	Ay
Area1	61.37	281.69	364.98	17287.32	22398.82
Area2	107.61	276.78	372.39	29784.3	40072.89
Area3	363.97	262.69	344.86	95611.28	125516.87
Area4	252.42	253.66	338.59	64028.86	85466.89
Area5	1175.62	266.1	295.75	312832.48	347689.62
Total	1960.37			519544.24	621145.09

$$X_{cm} = \frac{\sum Ax}{\sum A} = 265; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 316.86$$

e)

Area Name	Area	x	y	Ax	Ay
Area1	61.37	285.69	232.64	17532.8	14277.12
Area2	107.61	293.1	237.55	31540.49	25562.76
Area3	363.97	265.56	251.64	96654.05	91589.41
Area4	252.42	259.29	260.68	65449.98	65800.85
Area5	1175.62	216.46	248.23	154474.71	291824.15
Total	1960.37			465652.03	489051.19

$$X_{cm} = \frac{\sum Ax}{\sum A} = 237.53; \quad y_{cm} = \frac{\sum Ay}{\sum A} = 249.47$$

4.1.2 CAD computation of center of mass

There are three type of COM in Solidworks. COM, COM part and COM subassembly indicates global COM of entire model, COM of component in part file and COM of component in the subassembly file. All types of COM can be changed by selecting Override COM and specifying values for X, Y and Z coordinates. As a result, symbol of COM modifies its appearance as user-defined. It can be also created Center of Mass Reference (COMR) only in parts. COMR is a reference point at the current center of mass of the part. It can be added more feature to the part, global COM moves, but COMR point remains where it is created. It may change its position if features above COMR point are modified. It is also possible to create measurements between COM and COMR. It can be optimized location of COM in Design Study, for example: by adding extrusion or extruded cut. Information provided above gives to choose among COM types in order to explore properties of COM. The best trade off among three types of COM is COM part since dimensional and positional characteristics of elements in the metal sheet is not suitable for COM global and COM subassembly. In order to get COM coordinates of each element, extruded part is created in Solidworks part file in the figure 25 with default value along z axis and four edges of metal sheet is removed. Otherwise, shapes appear as hollow in part file. We are interested in more (centroids of area) x and y coordinates of center of mass comparing to z axis, since metal sheet has uniform density, so value along z is constant. In this file default value is used for density.

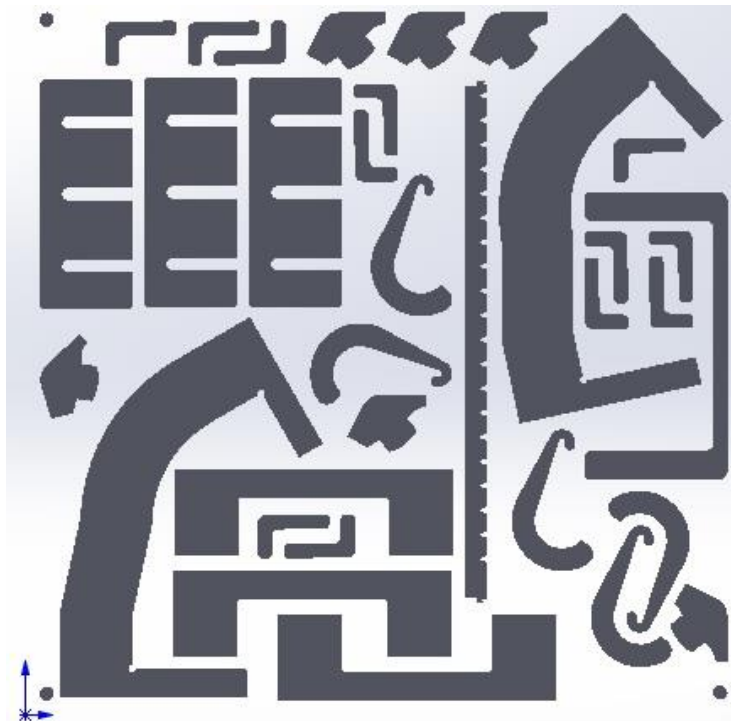

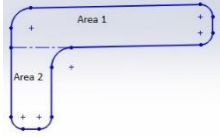
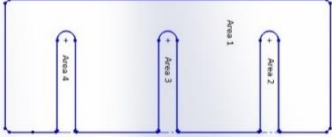



Figure 25. Extruded CAD version of work piece

Centroids of each element can be determined by two ways Mass properties and Section properties functions of Solidworks. All elements are created in single part file; this represents center of mass of whole system. It is desired to get COM of all elements at the same time, but Solidworks part file is not capable to represent such feature. However, it can be taken centroid coordinates of each element though choosing each element. Mass and Section properties functions do not only provide COM, but also show inertial properties in 3D and in 2D, respectively. In this context centroid and COM are used interchangeably.

4.1.3 Obtained center of mass results

N ^o	Figures	Manual computation of COM in (x, y)	COM generation in Solidworks	($\Delta x, \Delta y$)
1		a) (273.26, 34.66) b) (200.26, 75.76) c) (201.26, 146.59) d) (452.86, 264.48)	a) (273.26, 34.72) b) (200.26, 75.76) c) (201.26, 146.59) d) (453.00, 264.48)	a) (0, 0.06) b) (0, 0) c) (0, 0) d) (0.14, 0)
2		a) (211.16, 118.73) b) (181.35, 130.08) c) (74.77, 473.19) d) (132.76, 475.54) e) (429.7, 392.57) f) (162.57, 462.78) g) (250.64, 421.05) h) (239.62, 391.24) i) (411.35, 318.04) j) (400.33, 288.23) k) (456.59, 318.76) l) (445.57, 288.95)	a) (211.00, 119.03) b) (181.51, 130.25) c) (74.93, 474.35) d) (132.92, 474.30) e) (429.86, 392.75) f) (162.41, 463.07) g) (250.34, 420.90) h) (239.12, 391.40) i) (411.35, 318.04) j) (399.84, 288.39) k) (456.30, 318.60) l) (445.07, 289.11)	a) (-0.16, 0.3) b) (0.16, 0.17) c) (0.16, 0.16) d) (0.16, 1.24) e) (-0.04, 0.18) f) (-0.16, 0.29) g) (-0.3, 0.85) h) (-0.4, 0.16) i) (0,0) j) (-0.49, 0.16) k) (-0.29, -0.16) l) (-0.5, 0.16)
3		a) (41.03, 363.14) b) (114.03, 364.48) c) (187.03, 364.48)	a) (41.57, 363.24) b) (114.57, 364.48) c) (187.57, 364.48)	a) (0.54, 0.1) b) (0.54, 0) c) (0.54, 0)
4		a) (314.19, 260.24)	a) (314.17, 259.79)	a) (-0.02, -0.45)

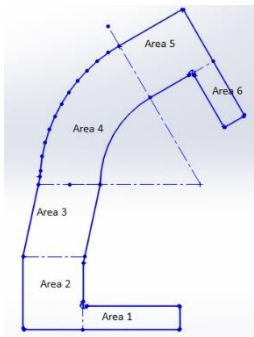
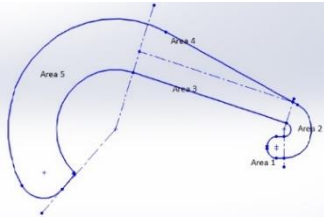
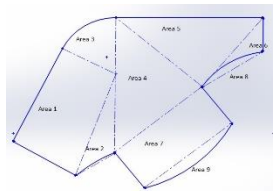
5		a) (90.96, 142.9) b) (379.86, 345.02)	a) (90.86, 141.89) b) (382.80, 343.86)	a) (-0.1, 1.01) b) (2.94, -1.16)
6		a) (363.33, 139) b) (439.88, 117.94) c) (418.6, 72.08) d) (265, 316.86) e) (237.53, 249.47)	a) 362.70, 138.65 b) 440.55, 118.14 c) 417.78, 71.65 d) 264.29, 316.40 e) 237.11, 250.04	a) (-0.63, -0.35) b) (0.67, 0.2) c) (-0.82, -0.43) d) (-0.71, -0.46) e) (-0.42, 0.57)
7		a) (279.79, 472.24) b) (337.23, 472.24) c) (223.52, 472.24) d) (252.51, 205.21) e) (30.17, 233.86) f) (472.24, 64.34)	a) (279.67 472.08) b) (337.11 472.08) c) (223.39, 472.08) d) (252.38 205.35) e) (30.19 233.66) f) (472.08 64.46)	a) (-0.1, -0.16) b) (-0.12, -0.16) c) (-0.13, -0.16) d) (-0.13, 0.14) e) (-0.02, -0.2) f) (-0.16, 0.12)

Table 4. Manual and Solidworks COM results

Table 4 represents manual and Solidworks computation of COM of seven different components and difference between them. Absolute values of difference are in the range between 0 and 2.94 along x axis and between 0 and 1.16 along y axis. The greatest difference can be seen on the fifth element on the table. There are 2 such kind of elements. The second one results the difference 2.94 and 1.16 along x and y axis, respectively, while the first one has an outcome more or less similar with the rest elements. Possible source of this greater mismatch can be making small assumptions on the first element and not applying for the second one, for example: neglecting some tiny parts. It may also happen making small mistakes choosing centroids for commonly known figures because of different orientation. Each element, except last one on the table, is computed separately even though they are the same. If element is complex shape and divided more areas, small errors may have cumulated. One possible solution is copying COM points directly to the other the same elements, once fair computation is achieved. This solution is applied to the last element on the table. Consequences with the other elements are quite reasonable since most of them is less even one. It is interesting to note that difference is not the same in both computation and imitation of COM for the same elements if they don't lie on the same line.

Results indicates COM of four types of components (sample 1, sample 3, sample 5 and sample 6) are outside their shape, while COM of one (sample 2) is located at the edge of polygon. Studies show COM parameter is not appropriate candidate for grasping point of components.

4.2 Grasping point along principal axes

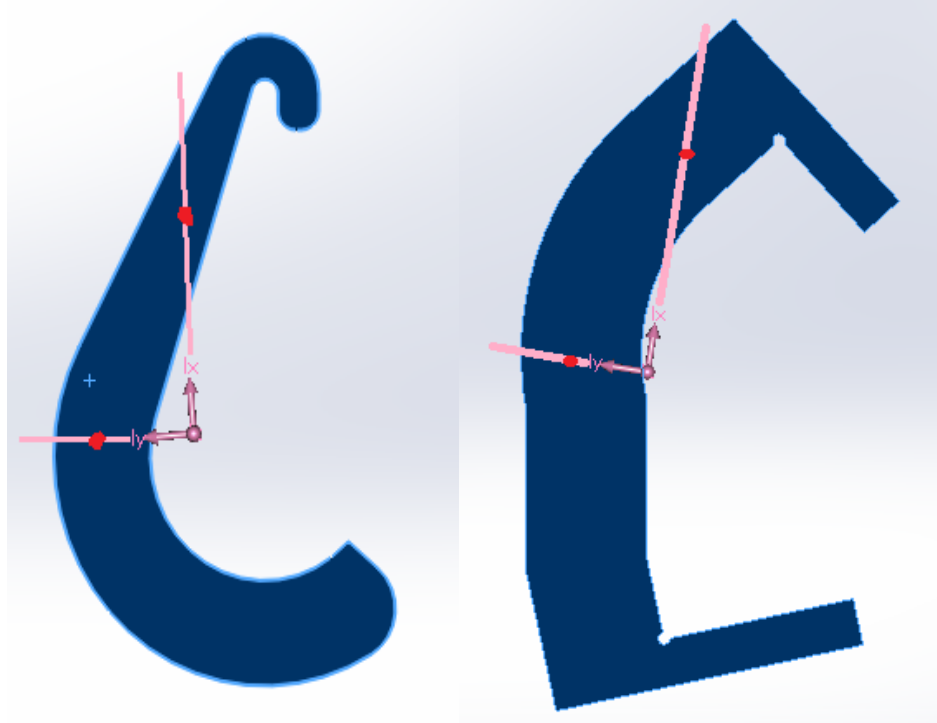


Figure 26. Principal inertial axes of sample 5 and sample 6.

COMs of these two samples from our work piece are located outside the shape and principal axes are generated in Solidworks using mass properties in the figure 26. It is not obvious orientation of principal axes of non-symmetric shape at a first glance. It can be seen Mass properties section of Solidworks. Once principal axes and their orientation are known, it can be drawn two line from COM along either x principle axis and y principle axis. It should be investigated some scenarios to choose grasping point candidate along one of principle axes.

- Scenario A

It can be selected a principle axis, that is closer to the edge of polygon, from two principal ones. Grasping point may be an intersection of edge and principle axis. In this case, it is recommended to add offset inward while robot tool center point is moving grasping point.

- Scenario B

The principle axis is the one closer to the edge. Principal axis line intersects with two edges of shape, forms a distance between intersections. Midpoint (red) of distance can be a candidate for grasping point.

- Scenario C

Principal axis is the one further to the edge of shape. Grasping point is an intersection of edge and principle axis. Inward offset should be applied.

- Scenario D

Principal axis is the one further to the edge of shape. Principal axis line intersects with two edges of shape, forms d distance between intersections. Midpoint (red) of distance can be a candidate for grasping point.

These scenarios are applicable only COM is outside the shape. If COM is inside the polygon, COM is a grasping point. Components under our studies are relatively small compared to the cross section of common magnetic grippers that are available in the market. This may cause problem while magnetic gripper is hold and detaching the samples. In order to solve this issue to some extent, it is desired to reach the grasping point where it is located at the most distant point from the edges of shape.

4.3 MIC implementation using Voronoi diagram in Matlab

The most distant point, from edges of polygon, is located at the Medial Axis of polygon. Thanks to Voronoi diagram, Voronoi nodes represent the Medial Axis of any convex polygon. It simplifies computation by reducing all nodes inside polygon to only Voronoi nodes of polygon. Suggested MIC algorithm using Voronoi diagram provide this desired solution.

Implementation of MIC algorithm using Voronoi diagram in Matlab are given in detail. Initial phase is an image pre-processing that is responsible for conversion correct type of image, noise reduction, removal of non-necessary shapes. The second phase is an implementation of the algorithm to find MIC. According to application of developed MIC solution, image type is required in the form of binary. Metal sheet image is truecolor in PNG file format in the figure 26. There is no direct conversion from truecolor to binary type. However, it is possible to convert grayscale to binary. RGB image first is required to convert it to grayscale and binary one. This is one of the solutions among other types of conversions. Formed binary image represents a change between objects and background, it leads to convert background to foreground. It may happen unwanted connections among components. That's why it is better to remove noise in the image by specifying threshold with `bwareaopen` function. Removing noise in this case is related with all connected components (objects) that have fewer than threshold pixels. On the other side there are three circles that are very small comparing to other shapes and not necessary for our analysis. It is also needed to remove them. Image Processing Toolbox provides circle detection function, unfortunately it does not work since pixels of circle are not accurate in our image. Another approach is a comparison area of shapes with threshold, shapes (circles) less

than threshold removed. It may happen the size of image is not desired and it is solved in Matlab by resize function in order to achieve 500mm x 500mm. Dimension of resize function is in pixel. That's why millimeters converted into pixels (1890x1890 in pixels). At the end of image pre-processing it is desired to get outline of all shapes in white/true/1, the other pixels must be in black/false/0.

At the end of Matlab script it is desired to get x and y coordinates of center MIC in millimeters. As we know script expresses results in pixels. The obtained results in pixel should be multiplied by 0.2645 and it forms in millimeters. It also should be considered y axis values increases from top to the bottom in image processing, unlike the Cartesian coordinate system. It is possible to solve this with simple math, while representing results in the Cartesian coordinate system.

Proposed solution is not only applicable for PNG image format , but also implimentable for BMP one. MIC algorithm is the same for both image formats, it differs only image pre-processing depending on specificatons of image formats.

3.2.1 MIC algorithm steps

1. Identify number of polygons N
2. Identify the boundary coordinates (x, y) of polygon
3. Make a Voronoi diagram (vx_i, vy_i) from boundary coordinates
4. Find Voronoi nodes (Vx_i, Vy_i) inside the polygon
5. Define minimum distance and minimum distance index: minDist=0 and minDistInd=-1
6. Evaluate max-min function
 - a) $r = \min_{\{i\}} r_{\{i\}}$ minimizes distance from each Voronoi node to all edge nodes of polygon, where $r_i = \|(Vx_i, Vy_i) - (x, y)\|$
 - b) Maximize r
 - If (r>minDist)
 - minDist =r;
 - minDistInd=i;
 - end
7. Return maximum radius and its coordinates: minDist, and Vx(minDistInd), Vy(minDistInd)
8. Go to step 2, until counter of polygons is N
9. The algorithm stops when counter of polygons is N+1

Full algorithm script is written in Matlab and given in the appendices section of thesis.

4.3.2 Obtained MIC results

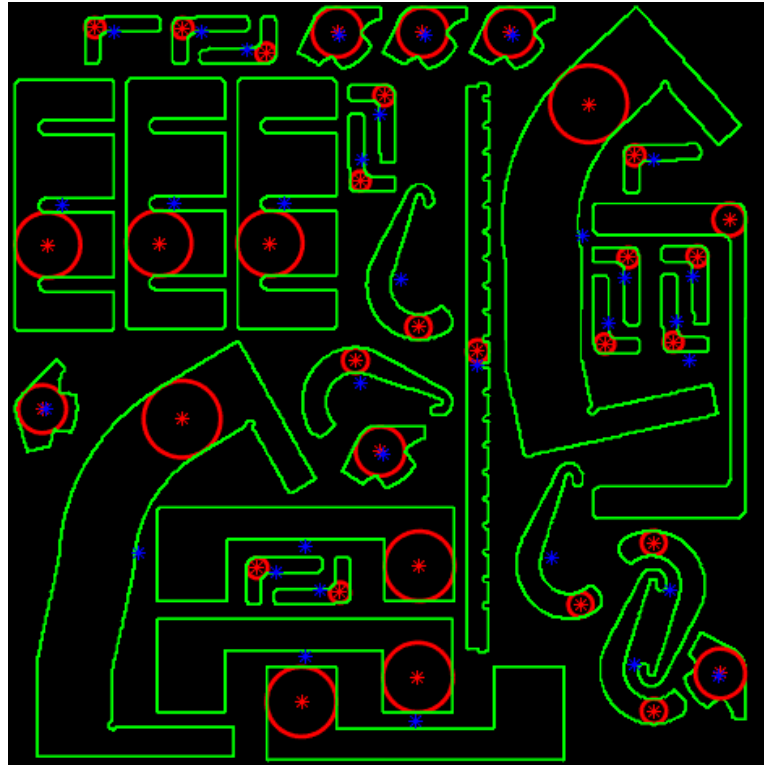


Figure 27. MIC simulation result in Matlab

The obtained results are given numerically in the table 5 and graphically in the figure 27. Solution is applicable for all elements of metal sheet under our investigation. MIC and origin of MIC is plotted in red while centroids are represented in blue. Origin of MIC coordinates are good enough for grasping points. Those points are within a polygon and are located equidistant between at least two edges of each polygon.

№	Figures	MIC centroids in Matlab in mm
1		a) (268.47, 131.32) b) (267.41, 58.59) c) (191.50, 42.72) d) (470.53, 357.59)

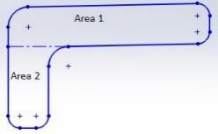
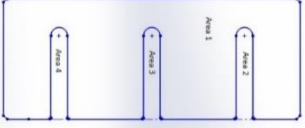

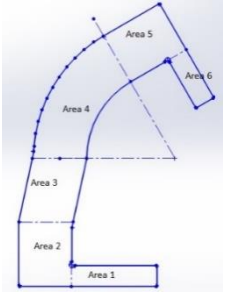
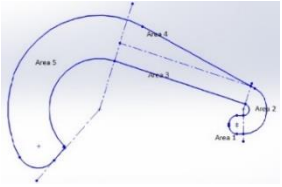
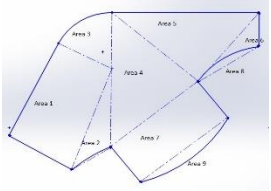
2		<p>a) (56.87, 482.97) b) (114.11, 482.69) c) (168.39, 466.41) d) (433.97, 277.65) e) (162.68, 130.16) f) (216.58, 114.05) g) (245.49, 438.72) h) (229.86, 382.74) i) (404.63, 332.79) j) (389.29, 276.44) k) (408.74, 399.57) l) (449.76, 333.68)</p>
3		<p>a) (26.05, 341.07) b) (98.53, 342.13) c) (171.00, 342.13)</p>
4		<p>a) (306.42, 271.77)</p>
5		<p>a) (113.73, 227.36) b) (378.83, 433.19)</p>
6		<p>a) (226.68, 265.61) b) (267.81, 287.38) c) (373.47, 106.10) d) (421.22, 146.53) e) (421.35, 36.62)</p>
7		<p>a) (22.48, 233.91) b) (214.75, 480.04) c) (270.56, 480.04) d) (327.43, 480.04) e) (242.89, 206.38) f) (464.65, 61.17)</p>

Table 5. MIC Centroids

5.Implementation of MIC results to Universal Robot 3 (UR3)

Next aim is making real experiments with UR3 in order to investigate only capability of reaching obtained MIC results in Cartesian coordinate system. This robot is chosen to make prototype for detaching metal sheet components since it is available in the laboratory of Politecnico di Torino.

5.1 Overview of UR3.

Current UR3 [13] belongs to CB series and has 3kg payload as well as 500 mm extension range from its base capabilities in the figure 28. There are other versions of Universal Robot such that UR5, UR10 that allow more payload and extension due to mechanical structure, UR3 is 6-axis robotic arm and its workspace is spherical.



Figure 28.UR3

5.1.1 Limitations of UR3

Metal sheet work piece is 50 cm x 50 cm. The work piece is placed as much as possible close to the base of UR3. Origin of working plane is located on the left of bottom of work piece, the same as done on the image processing. The shape and height of work piece that affects shrinkage task space of robot. It is obvious that TCP of robot can reach its maximum point when work piece is at the same height with base and is inside of operating circle/semi-circle with 50 cm radius on the plane. In our case, work piece is placed such that 8 cm higher from the origin of base and size of work piece exceeds from robot TCP operating circle radius.

5.1.2 UR 3 programming

Control box of UR3 uses operating system called Polyscope, based on Linux. Input and output interactions of UR3 are performed using one of USB, Ethernet, modbus, digital and analog I/Os. A teach pendant touch screen connected with the control box via cable allows to program the robot.

The robot has three programming options:

- The first one: The URCap is the language used on the teach pendant with GUI (graphical user interface).
- The second one: The URScript [14] is the robot programming language at the Script level.
- The third one is a combination of both options

5.1.3 Robot client interface selection

UR3 robot can interact with external devices through different types of communication interfaces [15] such as Primary/Secondary Interfaces, Real-time Interfaces, Dashboard Server, Socket Communication, XML-RPC and RTDE (Real-Time Data Exchange). Among those interfaces socket communication is chosen based on its specifications and requirements of system. TCP/IP protocol provides communication between UR3 and external device. UR3 acts as client and computer plays a role as server. URScript is capable to send and receive different data formats.

5.2 Workflow of MIC from Matlab to UR3

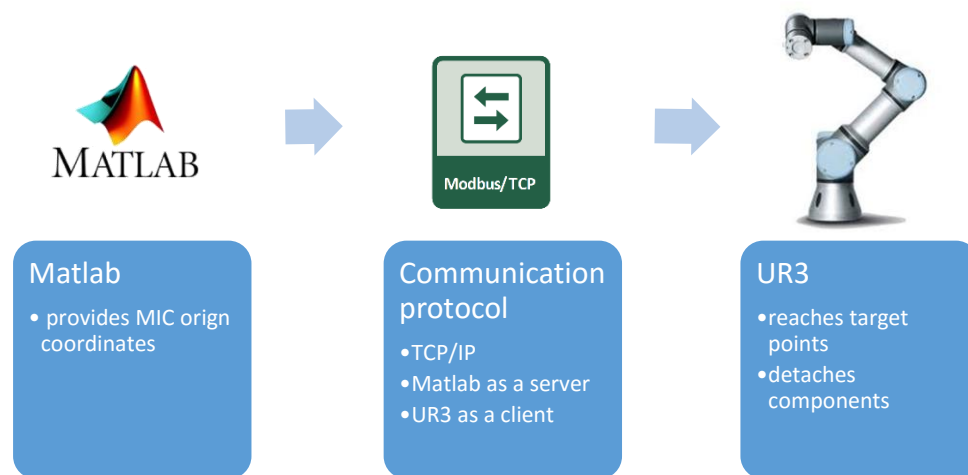


Figure 29. Workflow diagram

Matlab [16] computes MIC origin coordinates in 2D and stores them into two arrays such that x_coor and y_coor . It should be noted that computation of MIC origin coordinates is a bit time consuming. That's why it is recommended to store all elements into arrays, then to send them to UR3. Otherwise, time is not synchronized with UR3 and error occurs.

Once x_coor and y_coor arrays are ready, Matlab starts to open socket connection after receiving "yes" command from operator. Any data are converted into string and sent to the UR3 since it is a requirement of Polyspace. In the first step, number of sheet metal components are sent. This number

is assigned as maximum value of counter of the loop. In each iteration of the loop, it is read x and y coordinates from Matlab, separately. These coordinates forms target points of UR3.

Magnetic gripper of UR3 moves to the target, then it activates magnet and holds sheet metal sample, then detached components are put to appropriate place. At the end of each iteration gripper returns its reference point. Those steps are repeated until number of iterations reaches its maximum value. In this stage of thesis, gripper reaching to target points is studied.

Another matlab script is written for XML file of Libellula software. Libellula is specialized for nesting sheet metal components, provides options to save files in the forms of XML, BMP and DXF. Matlab script is intended to extract grasping point data from XML file and send them to robot.

5.2.1 Testing MIC results on UR3

Some tests are performed. UR3 works smoothly on the plane between ranges $0 < x < 50$ cm and $5.859 < y < 33$ cm. Total number of UR3 reachable pair coordinate points are 17 out of 33.

6.Applications of MIC results for samples with micro-joints

Micro-joints allow to attach components to sheet metal and they are formed by not cutting a minimal part of contour. The number and location of micro-joints are identified by considering shape, material size and thickness of component. Current softwares in the field of nesting allow to put micro joints manually and automatically. Both types of micro-joint placement are intended to hold components on the metal sheet. Algorithms of automated micro-joint placement of softwares are not known for different reasons. As much as less number of micro-joints are desired to have on the components in order to simplify detaching process. Most all of components have only one micro-joint, only sample 4 has two micro-joints in our work piece. It should be noted most of micro-joints are located at the edge of component along COM. Obtained MIC origin points are located at different from COM. If grasping points are aligned with principal axis, less detaching torque is required. If MIC origin points are treated as grasping points a bit more torque may be needed. Possible detach process of components with micro-joints can be as following:

- Tool center point reaches to grasping point
- Magnetic gripper is activated to hold the component
- Force is applied under certain degree up and down several times
- Detached component is put needed place

7. Conclusions

In this thesis, the issue of identification of grasping points of metal sheet components has been studied. The problem was geometrical centroids of work pieces being at the edge or outside of polygons and, as a result robot has a holding issue with components. Existing solutions in different fields and feasibility of their implementation are discussed. Importance and role of COM, principal axes and Medial Axis are mentioned theoretically and practically. COM and principal inertia axes of sample work piece are analyzed Solidworks. Limitations of Solidworks for our investigation leads to change into Matlab. MIC using Voronoi diagram is chosen as a solution and it is implemented in Matlab. Simulations are performed via Image Processing Toolbox of Matlab. Simulation MIC results are satisfied all elements of samples of work piece and as well as other different polygons without holes. Matlab simulation outcomes and UR3 cobot input/output interactions have been examined. Some tests are performed to check robot reaching target MIC origin points successfully.

Automation of detaching with micro-joints has still interesting topics for investigations. One very crucial of them is micro-joint crack. Future work can be micro-joint crack modelling by applying torque.

8. Bibliography

- [1] Daniel Garcia-Castellanos, Umberto Lombardo, “Poles of Inaccessibility: A Calculation Algorithm for the Remotest Places on Earth”, *Scottish Geographical Journal* Vol. 123, No. 3, 227 – 233, September 2007
- [2] Vladimir Agafonkin, “A new algorithm for finding a visual center of a polygon”, <https://blog.mapbox.com/a-new-algorithm-for-finding-a-visual-center-of-a-polygon>
- [3] Hoseok Kang, Shoreh Elhami, “Using Shape Analyses for Placement of Polygon Labels”, <https://proceedings.esri.com/library/userconf/proc01/professional/papers/pap388/p388.htm>
- [4] Burak Beyhan, Cüneyt Güler, Hidayet Tağa, “An algorithm for maximum inscribed circle based on Voronoi diagrams and geometrical properties”, *Journal of Geographical Systems*
- [5] Tolga Birdal, “Maximum Inscribed Circle Inside Polygon”, <http://www.mathworks.com/matlabcentral/fileexchange/30805-maximum-inscribed-circle-using-distance-transform>
<https://nl.mathworks.com/matlabcentral/fileexchange/32543-maximum-inscribed-circle-using-voronoi-diagram>
- [6] Andy Ruina and Rudra Pratap, “Introduction to Statics and Dynamics”, Chapter 2, pages 78-86
- [7] Stefan Huber, “Computing Straight Skeleton and Motorcycle Graphs”, PhD thesis, 2011
- [8] L. Lam, S.-W. Lee, and C. Y. Chen, “Thinning methodologies—A comprehensive survey”. *IEEE Sep*, 1992
- [9] Marek Teichmann, Seth Teller MIT, “Polygonal Approximation of Voronoi Diagrams of a Set of Triangles in Three Dimensions”, MIT Press.
- [10] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. W. Zucker, “The Hamilton-Jacobi skeleton”. In *International Conf. Computer Vision*
- [11] C.-S. Chiang, “The Euclidean distance transform”. Ph.D. thesis
- [12] J.-D. Boissonnat, “Geometric structures for three-dimensional shape representation”.
- [13] Universal Robot site: www.universal-robots.com
- [14] Universal Robot, “The URScript Programming Language”, Version 3.4.5, 2017
- [15] Universal Robot file for client interface information, “Client_Interface_V3.5.xlsx”
- [16] Mathwork site: www.mathwork.com
- [17] Rajesh Ramamurthy, Rida T. Farouki “Voronoi diagram and medial axis algorithm for planar domains with curved boundaries I. Theoretical foundations “, *ournal of Computational and Applied Mathematics* 102 (1999) 119-141

9. Appendices

Matlab script for png format

```
%% Creating connection with Matlab-UR3

%%% Implementation of MIC algorithm via image processing
clear all
close all
clc

I= imread('dimension_in snipper.png');
imshow(I)
J = imresize(I,[1890 1890]);
imshow(J)

I2 = rgb2gray(J);
imshow(I2)
bw = imbinarize(I2); % function to convert the grayscale image into a binary
image
bw = bwareaopen(bw,100); % Remove background noise from the image with the
bwareaopen function.
bw = ~bw; % change background to foreground
imshow(bw)

% removing small, non-interesting shape (three circles)
CC = bwconncomp(bw, 8);
S = regionprops(CC, 'Area');
L = labelmatrix(CC);
BW2 = ismember(L, find([S.Area] > 1136));
imshow(BW2)

b=bwboundaries(BW2); % bw is your binary image you want to operate on

binaryMask=false(size(BW2));
for i = 1:length(b) % this functon is for filling within shape with black
    for j = 1:length(b{i})
        ind = b{i}(j,:);
        binaryMask(ind(1),ind(2))=1;
    end
end

imwrite(binaryMask,'dim_snipp.png')

disp('Uploading image ')

Iorg=imread('dimension_in snipper.png');
I=imread('dim_snipp.png');
imshow(I)

[B,L] = bwboundaries(I,'noholes'); % obtain the boundary
```

```

for k = 1:length(B)
    contour = B{k}; % obtain contour of each polygon

    [y,x]=find(I>0);

    % get the circle
    [cx,cy,r]=find_inner_circle(contour(:,2), contour(:,1));

    % plot
    theta = [linspace(0,2*pi) 0];
    hold on, plot(x,y, 'g.', 'MarkerSize',0.8);
    hold on, plot(cos(theta)*r+cx,sin(theta)*r+cy, 'r', 'LineWidth', 2);

    % getting coordinates of each polygon P(x,y)
    x_y_coordinates(k,:)=[round(cx*0.2645,2)/10; round((1890-
cy)*0.2645,2)/10];

end

    x_coor=x_y_coordinates(:,1).';
    y_coor=x_y_coordinates(:,2).';

%%% sorting MIC results to TCP achievable one
number_of_objects=length(B);
j=1; % counter initialization
    for i=1:number_of_objects
        if (x_coor(1,i)<50) && (y_coor(1,i)<33)
            x_c(1,j)=x_coor(1,i);
            y_c(1,j)=y_coor(1,i);
            j=j+1;
        end
    end

%%% Socket connection of UR3 and Matlab
Robot_IP = '192.168.12.50'; %Enter Robot IP
robot = tcpip(Robot_IP,30003,'NetworkRole','server');
fclose(robot);

disp('Press Play on Robot...')
fopen(robot);
disp('Connected to Robot!');
disp('Ready...?, YES/NO')
ready = input('Ready to start? (y/n): ','s');
C=1;
if strcmp(ready,'y')
    disp(ready)
else
    C=0;
    fprintf(robot,'(0)');
    fclose(robot);
    disp('The Socket is closed! Restart program!')
end
end

```



```

%% Sending number of objects and x, y coordinates to UR3
j=j-1; % number of objects below 35
n=num2str(j);
fprintf(robot, '%s\n', ['(',n,')']);
k=1; % counter initialization

while (k<j)

    x=num2str(x_c(k));

    fprintf(robot, '%s\n', ['(',x,')']);
    pause(0.5)

    y=num2str(y_c(k));
    fprintf(robot, '%s\n', ['(',y,')']);
    k=k+1;

end

% given a polygon [x,y] find the inner circle [cx,cy,r]
function [cx,cy,r]=find_inner_circle(x,y)

% make a voronoi diagram
[vx,vy]=voronoi(x,y);

% find voronoi nodes inside the polygon [x,y]
Vx=vx(:);
Vy=vy(:);
% Here, you could also use a faster version of inpolygon
IN=inpolygon(Vx,Vy, x,y);
ind=find(IN==1);
Vx=Vx(ind);
Vy=Vy(ind);

% maximize the distance of each voronoi node to the closest node on the
% polygon.
minDist=0;
minDistInd=-1;
for i=1:length(Vx)
    dx=(Vx(i)-x);
    dy=(Vy(i)-y);
    r=min(dx.*dx+dy.*dy);
    if (r>minDist)
        minDist=r;
        minDistInd=i;
    end
end

% take the center and radius
cx=Vx(minDistInd);
cy=Vy(minDistInd);
r=sqrt(minDist);

end

```

Matlab script for bmp format

```
clear all;
clc

%%% Conversion rgb image into gray scale one and resize it
[X,map] = imread('bitmap_format.bmp');
newmap = rgb2gray(map);
imshow(X,newmap)

bw = imbinarize(X); % function to convert the grayscale image into a binary
image
bw = bwareaopen(bw,50); % Remove background noise from the image with the
bwareaopen function.
X2= imresize(bw,[1890 1890]);

imshow(X2)

imwrite(X2,'binary.bmp')

I=imread('binary.bmp');
imshow(I)

% obtain the boundary
[B,L] = bwboundaries(I,'noholes');

% obtain contour of each polygon
for k = 1:length(B)
    contour = B{k};

    [y,x]=find(I>0);

    % get the circle
    [cx,cy,r]=find_inner_circle(contour(:,2), contour(:,1));

    % plot
    theta = [linspace(0,2*pi) 0];
    hold on, plot(x,y, 'g.', 'MarkerSize',0.8);
    hold on, plot(cos(theta)*r+cx,sin(theta)*r+cy,'r', 'LineWidth', 2);
    hold on
    % getting coordinates of each polygon P(x,y)
    x_y_coordinates(k,:)=[round(cx*0.2645,2); round((1890-cy)*0.2645,2)];
end

%%% sorting MIC results to TCP achievable ones
x_coor=x_y_coordinates(:,1).';
y_coor=x_y_coordinates(:,2).';
number_of_objects=length(B);
j=1; % counter initialization
for i=1:number_of_objects
    if (x_coor(1,i)<50) && (y_coor(1,i)<33)
        x_c(1,j)=x_coor(1,i);
        y_c(1,j)=y_coor(1,i);
        j=j+1;
    end
end
end
```

```

%% Socket connection of UR3 and Matlab
Robot_IP = '192.168.12.50'; %Enter Robot IP
robot = tcpip(Robot_IP,30003,'NetworkRole','server');
fclose(robot);

disp('Press Play on Robot...')
fopen(robot);
disp('Connected to Robot!');
disp('Ready...?, YES/NO')
ready = input('Ready to start? (y/n): ','s');
C=1;
if strcmp(ready,'y')
    disp(ready)
else
    C=0;
    fprintf(robot,'(0)');
    fclose(robot);
    disp('The Socket is closed! Restart program!')
end

%% Sending number of objects and x, y coordinates to UR3

j=j-1; % number of objects below 35
n=num2str(j);
fprintf(robot,'%s\n',['(',n,')']);
k=1; % counter initialization

while (k<j)

    x=num2str(x_c(k));

    fprintf(robot,'%s\n',['(',x,')']);

    y=num2str(y_c(k));
    fprintf(robot,'%s\n', ['(',y,')']);
    k=k+1;

end

% given a polygon [x,y] find the inner circle [cx,cy,r]
function [cx,cy,r]=find_inner_circle(x,y)

% make a voronoi diagram
[vx,vy]=voronoi(x,y);

% find voronoi nodes inside the polygon [x,y]

```

```

Vx=vx(:);
Vy=vy(:);
% Here, you could also use a faster version of inpolygon
IN=inpolygon(Vx,Vy, x,y);
ind=find(IN==1);
Vx=Vx(ind);
Vy=Vy(ind);

% maximize the distance of each voronoi node to the closest node on the
% polygon.
minDist=0;
minDistInd=-1;
for i=1:length(Vx)
    dx=(Vx(i)-x);
    dy=(Vy(i)-y);
    r=min(dx.*dx+dy.*dy);
    if (r>minDist)
        minDist=r;
        minDistInd=i;
    end
end

% take the center and radius
cx=Vx(minDistInd);
cy=Vy(minDistInd);
r=sqrt(minDist);

end

```

Matlab script for xml format

```
clear all
clc

%%% x and y coordinates from xml file are stored to arrays in matlab
xmlDoc = xml2struct('test_khurshid_1.xml');
lngth=length(xmlDoc.NestingInfo.MachineList.Machine.PartList.ReferencePart); %
number of elements
for i=1:lngth

X(i)=str2num(xmlDoc.NestingInfo.MachineList.Machine.PartList.ReferencePart{1,
i}.GraspingPoint.Attributes.X);
Y(i)=str2num(xmlDoc.NestingInfo.MachineList.Machine.PartList.ReferencePart{1,
i}.GraspingPoint.Attributes.Y);

end

%%% mm to cm conversion
X=X/10;
Y=Y/10;

%%%selection detachable objects along y axis in the range between 0 and 35 cm
j=1;

    for i=1:lngth
        if (Y(i)<33)

                x(j)=X(i);
                y(j)=Y(i);
                j=j+1;

            end
        end

%%%TCP/IP Connection with UR3

Robot_IP = '192.168.12.50'; %Enter Robot IP
robot = tcpip(Robot_IP,30003,'NetworkRole','server');
fclose(robot);

disp('Press Play on Robot...')
fopen(robot);
disp('Connected to Robot!');
disp('Ready...?, YES/NO')
ready = input('Ready to start? (y/n): ','s');
C=1;
if strcmp(ready,'y')
    disp(ready)
else
    C=0;
    fprintf(robot,'(0)');
    fclose(robot);
    disp('The Socket is closed! Restart program!')
end
```

```
%%% Sending number of objects and x, y coordinates to UR3
len=length(x); % number of objects below 35
n=num2str(len);
fprintf(robot, '%s\n', ['(', n, ')']);

for k=1:len

    x_c=num2str(x(k));
    fprintf(robot, '%s\n', ['(', x_c, ')']);

    y_c=num2str(y(k));
    fprintf(robot, '%s\n', ['(', y_c, ')']);

end
```

