

# A Simulation of Automatic 3D Acquisition and Post-processing Pipeline

Arsalan MALIK, Benjamin LORIOT, Youssef BOKHABRINE, Patrick GORRIA and Ralph SEULIN

Le2i Laboratory - UMR CNRS 5158 - Université de Bourgogne  
IUT - 12 rue de la Fonderie - 71200 LE CREUSOT - FRANCE

## ABSTRACT

This paper presents a simulation of automatic 3D acquisition and post-processing pipeline. The proposed methodology is applied to a LASER triangulation based scanner and a 6 degrees of freedom (DOF) robotic arm simulation. The viewpoints are computed by solving a set covering problem to reduce the number of potential positions. The quality of the view plan is determined by its length and the percentage of area of the object's surface it covers. Results are presented and discussed on various shapes. The article also presents future work concerning the implementation of the proposed method on a real system.

**Keywords:** 3D digitization, automation, next best view, view planning

## 1. INTRODUCTION

The 3D models are used in many applications such as computer games, movies, medical, training simulators, augmented reality, archaeology and industrial inspection. The 3D models can be artificially generated by artistic or technical design process using any off-the-shelf modeling software. The artificially generated 3D models are sufficient for most of the entertainment industry such as computer games or movies. However, applications such as archaeology or non-destructive industrial inspection demand true representation of the real complex objects. A better representation of the real complex objects can be achieved by measuring real shape and photometric properties in a 3D digitization process.

The 3D digitization (or measurement) process can be divided into two general steps called *Acquisition* and *Post-processing*. The acquisition system generally comprises a 3D digitizer (e.g. LASER range scanner, Time of Flight or Stereo Vision) mounted on a positioning system (e.g. Articulated Arm). The post-processing step includes registration, merging, hole-filling, cleaning and photometric and environment mapping.

The 3D digitization process requires selection of different viewpoints. The viewpoints are usually selected by a specialized human operator. The quality of the final result depends on selection of viewpoints. The amount of 3D digitized data depends on number of viewpoints. The human operator selects overlapping viewpoints most of the time and as a consequence, the same region of the object is sampled many times. The post-processing cost increases as a result of redundant data. Therefore the efficiency of 3D digitization process dependant on operator expertise and not on quantitative and objective methods. Automating the 3D digitization process can improve efficiency and quality which not only reduces post-processing cost but also demands less of the operator skills. Currently there is no fully automated 3D digitization system available commercially, and there is a strong demand in the industry.

This article presents a preliminary work for the implementation of automatic 3D acquisition and post-processing on a real system. The feasibility of automatic 3D acquisition and post-processing for a real system is demonstrated by simulation. The simulated model-based view planning will be used by our team in the next future with a real 3D digitizer mounted on a positioning system.

---

Further author information: (Send correspondence to Ralph SEULIN)  
Ralph SEULIN: E-mail: ralph.seulin@u-bourgogne.fr, Telephone: +33 (0)3.85.73.11.25

## 2. RELATED WORK

The automatic 3D acquisition requires view or path planning. The view planning is a well studied problem and have many proposed solutions. A comprehensive description of *View Planning Problem* in the context of 3D digitization is given by Scott et al.<sup>1</sup> Another survey on this topic is given by Tarabanis et al.<sup>2</sup> The view planning can be classified into *model based* and *non-model based* methods. In model based view planning, a complete or partial model (CAD or Mesh) of the object is available. The view planning step is usually offline and interactive. In non-model based view planning, no information about object is available. The view planning is done during the acquisition process. The presented work concentrates on model-based view planning.

There are several model-based view planning solutions in the literature.<sup>1,2</sup> A CAD model based method of automatic sensor placement for robot vision in the inspection tasks is given by Chen and Li.<sup>3</sup> Their plan is evaluated by min-max criterion, which is achieved by *Hierarchical Genetic Algorithm* (HGA), and the shortest path for robot moving through the viewpoints is determined by *Christofides algorithm*.<sup>3</sup> Tarabanis et al.<sup>4</sup> have developed a model-based sensor planning system for robot vision called *Machine Vision Planner* (MVP). The MVP is capable of taking object geometry from CAD models, and determine sensor pose and settings for which object features are visible, resolvable, in focus and contained entirely in the sensor field-of-vision (FOV). The MVP takes *synthesis* approach to generate view plan that satisfies above constraints.<sup>4</sup>

Abrams et al.<sup>5</sup> have presented a dynamic sensor planning system for an active robot work cell. Their system is capable of planning the pose and settings of the sensors for use in an environment containing objects moving in known trajectories. The focus of their work is *surveillance planning*,<sup>5</sup> in which there are more than one stationary sensors, which can be activated at different intervals of times to ensure that different view planning constraints are met. In addition to object model, the motion model is also an input to the system. They have also extended the abilities of the original MVP<sup>4</sup> based on this approach.<sup>6</sup> Our work is concentrated on static objects.

Trucco et al.<sup>7</sup> have demonstrated a system called GASP (General Automatic Sensor Planning) for model-based planning of optimal sensor placement for the inspection task. The GASP uses *Feature Inspection Representation* (FIR) which is generated offline using CAD model of the object, to compute online plans called *inspection scripts*.

Prieto et al.<sup>8</sup> have proposed a CAD based 3D acquisition strategy for inspection. Their algorithm generates 3D voxel model from the given CAD model. The best viewpoint placement and non-occlusion conditions are tested on 3D voxel data. Martins et al.<sup>9</sup> also presented a CAD based automatic surface scanning using a voxel representation. They generate viewpoints and test them according to two criterions, i.e. *viewable* and *accessible*. They generate collision free scanning trajectory based on surface following strategy. The method is tested using an optical range finder mounted on a 5 DOF CMM. An overview of different surface following techniques for surface scanning is given by Pudney in his dissertation.<sup>10</sup>

Ailisto<sup>11</sup> describes the development of CAD model-based semi-automated 3D measurement system using CMM and LASER rangefinder. A measurement planning tool is developed which generates necessary information for the automatic measurement using geometric information from CAD model. The measurement planning is an interactive process, while actual measurement is automatic. He has demonstrated feasibility of the concept for entire operational chain from measurement planning to comparison between acquired measurement and reference model.<sup>11</sup>

Scott<sup>12</sup> presented a multi-stage model-based view planning. In first stage, a rough or exploratory model of the object is obtained using a pre-programmed viewpoints. This approximate model is used to plan viewpoints that can be used in second stage to construct high quality *fine model*. The rough model-based view planning algorithm is based on *Measurability Matrix* originally proposed by Tarbox and Gottschlich.<sup>13</sup> The *Measurability Matrix* is formulated as *Set Cover Problem* and solved by simple greedy heuristic. Scott also proposed two different model-based algorithms to generate viewpoint space representation.

A multi-stage view planning is also presented by Mehdi et al.<sup>14</sup> Their approach is different from Scott in that they first scan the object to acquire 3D points cloud, which they subsequently convert into a voxel model for *intelligent scan planning*.<sup>14</sup> The new scan paths for an optimal digitizing is generated in second stage which is based on analysis of data quality. The quality of the model is determined by computing the characteristic

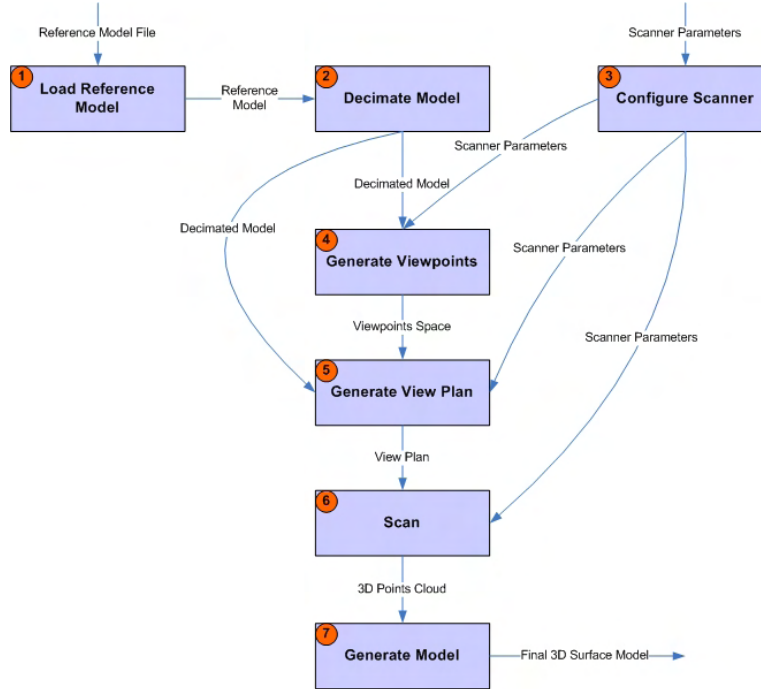


Figure 1. Simulation Pipeline

edges, unsatisfactory quality zones and holes. An experimental application on a CMM equipped with LASER plan sensor is presented. The model-based approach used in our work is based on Scott's method.<sup>12</sup>

### 3. SIMULATION PIPELINE

The simulation is implemented in RapidForm<sup>15</sup> and MATLAB. An overview of the simulation pipeline is given by the Fig. 1. The details of each step in the pipeline is explained below:

#### 3.1 Load Reference Model

The reference model is a high quality triangular mesh of the object to be digitized. The reference model is used as actual object to be digitized in the simulation.

#### 3.2 Decimate

The reference model usually has large number of vertices and faces (triangles). The space and time complexity of the view planning algorithm depends on the number of faces in the model used. A decimated copy of the reference model is made to reduce the number of faces and thus the complexity of the view planning step.

#### 3.3 Configure Scanner

A LASER Line triangulation scanner is modeled in the simulation. The scanner has two components, *source* and *sensor* as shown in the Fig. 5. The distance between the *source* and the *sensor* is called *baseline*. The scanner can be posed in any position and orientation in 6 DOF space. The frustum of the scanner can be configured using horizontal and vertical FOV angles and resolutions. Additionally the minimum and maximum range of the scanner can be specified. The *baseline* distance can be configured to model any bistatic scanner. The threshold angle is the maximum allowed *grazing angle*, which is expressed as an angle between the face normal and the source ray.

The frustum of the scanner is modeled as a spherical cone, with horizontal and vertical FOV angles as illustrated in the Fig. 2. The center of the sphere is at the origin in the scanner coordinate system. The two cylinders represent scanner's *source* and *sensor*. The lines represent selected rays within scanner's frustum. The scanning process is explained in the Section 3.6.

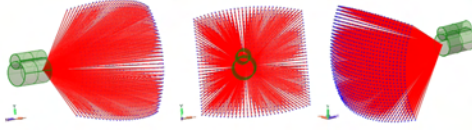


Figure 2. Different Views of Scanner Frustum Model

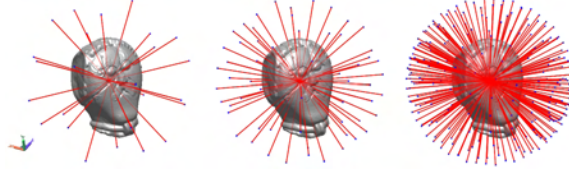


Figure 3. View Spheres with different Resolutions

### 3.4 Generate Viewpoints Space

The position and orientation of the 3D scanner is referred to as *pose* or *viewpoint*. The sampling of all possible viewpoints is called *viewpoints space*. The viewpoints space is generated using two different methods. They are *view sphere* and *optimal scanning zone*.

#### 3.4.1 View Sphere

This method is based on sampling of the view sphere centered on the object's center of mass with the appropriate radius that can encapsulate the entire object. The sampling is done using rotation angle resolution about x-axis and y-axis. The orientation of the scanner is always directed towards the center of the view sphere, i.e. center of mass of the object. The three view spheres with different resolutions ( $45^\circ$ ,  $30^\circ$  and  $15^\circ$ ) are illustrated in Fig. 3.

#### 3.4.2 Optimal Scanning Zone

This method uses the decimated object model to generate viewpoints space. It is based on optimal scanning zone algorithm presented by Scott.<sup>12</sup> For each face in the decimated model, the face normal is computed. Then the position of the scanner is selected at a fixed *standoff distance* from the center of the face towards the direction of face normal. The scanner orientation is selected in the direction opposite to the face normal. The number of viewpoints generated using this method is equal to the number of faces of the model used. The viewpoints space generated for an example model is shown in the Fig. 4.

#### 3.4.3 Positioning System Constraint

The Kuka KR16 industrial robot<sup>16</sup> is used as a positioning system in the simulation. This is a low payload industrial robot with 6 axes or 6 DOFs. The robot is mathematically modeled in *MATLAB Robotics Toolbox* by Corke.<sup>17</sup> The *Forward Kinematics* of the robot is modeled using standard *Denavit-Hartenberg* (DH) algorithm.<sup>18</sup>

Once the robot is mathematically modeled, a viewpoint can be tested to see if the desired pose can be achieved. This is done by solving *Inverse Kinematics* problem. The *Inverse Kinematics* is the process of determining the joint parameters of the robot in order to achieve a desired pose. If the desired pose cannot be achieved, there is no solution of the *Inverse Kinematics*. The *Inverse Kinematics* is solved for each viewpoint, and if there is no solution, the corresponding viewpoint is rejected, otherwise it is added in the viewpoints space.

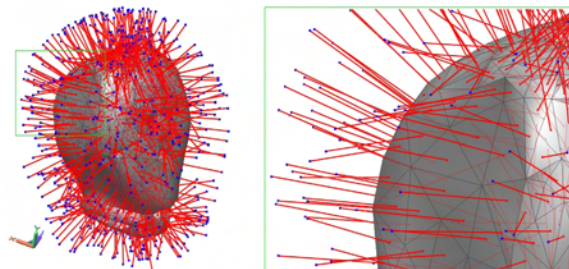


Figure 4. Viewpoints Space using Decimated Model

### 3.5 Generate View Plan

The view planning algorithm selects minimum number of viewpoints from *viewpoints space* that cover maximum surface of the object. The implemented algorithm is based on 3M algorithm presented by Scott.<sup>12</sup> The basic data structure is a measurability matrix  $M = [m_{ij}]$ . The columns of  $M$  represents viewpoints  $V$ , while rows represents surface features  $S$ . In this implementation, the individual faces (triangles) are used as surface features. The dimension of  $V$  is equal to number of viewpoints in viewpoints space, while dimension of  $S$  is equal to number of faces in decimated model. The view plan is generated in two steps:

#### 3.5.1 Fill M

Each element of  $M$  is a binary number. An element  $m_{ij}$  is 1 if the surface feature  $s_i$  is visible from the viewpoint  $v_j$ , otherwise its 0. The visibility of surface feature  $s_i$  is determined using scanner model. The scanner is positioned at a viewpoint  $v_j$  and the scanning process is executed. The surface features that are visible from  $v_j$  are identified and the corresponding elements of the column  $v_j$  are set as 1. The details of scanning process is given in the Section 3.6. The process is repeated for all  $v_j$  in  $V$ . The computational complexity of filling  $M$  is evident. It is in the order of  $V$  times the complexity of scanning one view. The computational complexity of scanning is given in the Section 3.6.

#### 3.5.2 Solve Set Covering Problem

The view planning problem can be formulated as *Set Covering Problem* (SCP).<sup>12,19</sup> SCP is, given several sets, that have many elements in common, the goal is to select minimum number of these sets that can cover maximum number of elements. In view planning context, given several viewpoints, the goal is to select minimum number of viewpoints that can cover maximum number of surface features. SCP is a classical NP hard optimization problem in a strong sense.<sup>20</sup> Several well established heuristics are available to solve this problem, including greedy, genetic, simulated annealing and Lagrangian relaxation.<sup>12</sup> A recent solution based on meta-heuristic Meta-RaPS (Meta-heuristic for Randomized Priority Search)<sup>20</sup> is used in our work.

### 3.6 Scan

The functionality of a LASER line triangulation based scanner is simulated. First, the frustum cone is generated for each pose. The frustum cone is represented by a sampling of rays originating from its center, that covers entire FOV. The number of rays is dependant on horizontal and vertical resolutions, and can be found by Eq. 1:

$$\text{Number of Rays} = \left( \frac{FOV_H}{\text{Resolution}_H} \right) \left( \frac{FOV_V}{\text{Resolution}_V} \right) \quad (1)$$

The actual scanning is simulated using *ray tracing* as illustrated by the Fig. 5. A ray (*active ray*) is traced from the scanner source towards its direction in the frustum. The first intersection point between the ray and a surface feature is selected if available. The distance between intersected point and scanner's source is computed. If the distance is outside scanner's range, the point is rejected, otherwise a second ray (*reflected ray*) is traced. The origin of second ray is the intersected point, and direction is towards the scanner sensor. If any intersection point is found, the point is rejected, because this condition corresponds to the *Shadow Effect*, otherwise the point is added to the points cloud set.

The above process is repeated for each ray in the frustum. Therefore the computational complexity of scanning one view is  $O(r)$  times the complexity of ray tracing, where  $r$  is the number of rays (see eq. 1). The frustum is generated for each viewpoint in the *View Plan*, or in the case of measurability matrix  $M$ , it is generated for all viewpoints in  $V$ . Thus, the computational complexity of scanning or filling  $M$  process is dependant on scanner FOV angles and resolutions as well as number of surface features in the model used.

### 3.7 Generate Model

The data obtained from scanning is a points cloud set. Since the object position and orientation is not changed during scanning in the simulation, the registration of different views is not required. The points cloud sets from all the views are merged into a single set. The surface triangulation converts points cloud set into a triangular mesh.

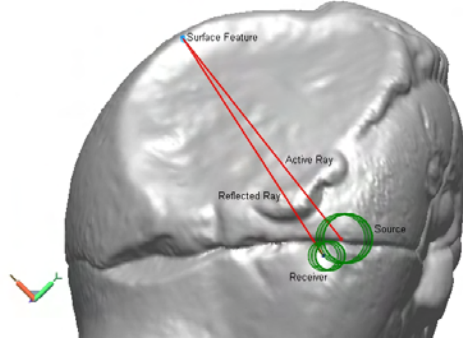


Figure 5. Ray Tracing Process in the Simulated Scanner

 <p><b>Mug</b> Vertices: 74994 Faces: 149988 Area: 16268mm<sup>2</sup> Bounds: 60 × 50 × 47mm</p>	 <p><b>Joueur Pétanque</b> Vertices: 169952 Faces: 339900 Area: 62757mm<sup>2</sup> Bounds: 207 × 179 × 74mm</p>	 <p><b>Robinet</b> Vertices: 81450 Faces: 162916 Area: 64345mm<sup>2</sup> Bounds: 228 × 120 × 108mm</p>
 <p><b>Mseke</b> Vertices: 93169 Faces: 186334 Area: 120737mm<sup>2</sup> Bounds: 188 × 223 × 183mm</p>	 <p><b>Humerus</b> Vertices: 138966 Faces: 277928 Area: 978mm<sup>2</sup> Bounds: 60 × 11 × 9mm</p>	 <p><b>St. Jean Baptiste</b> Vertices: 374995 Faces: 749994 Area: 29462mm<sup>2</sup> Bounds: 73 × 158 × 46mm</p>

Figure 6. Models used in the experiments

## 4. EXPERIMENTS AND RESULTS

Six different reference models are used in the experiments. The reference models are shown in the Fig. 6. The models have variations in terms of size and complexity. The model is complex when it has features such as holes, cavities or protrusions, which are difficult to digitize because they are not reachable by the 3D digitizer. The simplest model in the data set is **Mseke** because it does not have any complex features such as holes or perturbations. A more complex model is **Mug** which has a hole and a cavity. The most complex model is **Robinet**, which has lot of holes and cavities.

The experiments are done using different viewpoints spaces and scanner parameters. The results of each variation and its effect are given below. The results are evaluated based on the length of the view plan and percentage of the surface area it covers.

### 4.1 Viewpoints Space

The *Viewpoints Space* is generated using *Optimal Scanning Zone* algorithm as described in the section 3.4. The resultant viewpoints space can be changed using different decimation ratios and standoff distances. Moreover, it can be constraint using a positioning system model. The results of using positioning system constraint are given in the section 4.3. The results of using different decimation ratios and standoff distances are given below:

Table 1. Effect of Decimation Ratio

Model	Ratio	Viewpoints	Viewplan	Cover
Mseke	0.5	930	11	97.91
	1	1858	14	98.05
	2	3724	34	98.01
Humerus	0.1	276	6	95.55
	0.5	1386	16	95.43
	1	2776	38	95.46
Mug	0.1	148	6	81.68
	0.5	748	12	89.19
	1	1498	118	52.48
Robinet	0.5	814	19	87.60
	1	1628	40	89.14
	2	3256	83	88.33
Joueur Pétanque	0.1	338	8	94.02
	0.5	1698	29	94.90
	1	3398	67	94.36
St.Jean Baptiste	0.1	746	10	86.92
	0.2	1496	25	89.80
	0.5	3746	69	89.31

#### 4.1.1 Decimation Ratio

Decimation ratios between 0.1 and 2 percent are used in the experiments i.e. between 0.1 and 2 percent of the original points are kept. The results are given in the Table 1. The *Viewpoints* column gives number of viewpoints in the viewpoints space. The *Viewplan* column gives the length of the view plan. The *Cover* column gives the percentage of area covered by the view plan.

It can be seen in the Table 1 that if the decimation ratio is increased, the length of the view plan is also increased, while area covered is approximately the same. This demonstrated the ability of the algorithm to use a rough or coarse model for view planning.

A notable anomaly in the Table 1 is that for **Mug** with decimation ratio 1, the area covered is only 52.48 percent as compared to other results of 81.68 and 89.19 percents. Although the length of the view plan is large (118) as compared to other view plans (6 and 12), the actual number of points scanned by the scanner is significantly small due to *Shadow Effect*. This is a typical phenomenon when using bistatic scanner with such objects. It is also observed in the other results using **Mug** (Table 2), when standoff distance is 300 or more.

#### 4.1.2 Standoff Distance

The standoff distance is selected based on object bounds given in the Fig. 6. The results of using different standoff distances are given in the Table 2.

It can be seen in the Table 2 that there is an optimum standoff distance for each model, for which the length of the view plan is minimum and the area cover is maximum. For example, for **Humerus**, the optimum distance is between 30 and 50mm, while for **Joueur Pétanque**, the optimum distance is 300mm. If the standoff distance is outside the optimum zone, length of the view plan increases while area cover decreases. This is very evident in **Mug** results. The area cover is nearly 50 percent when standoff distance is 300 or more. This is due to the fact that **Mug** has a large cavity and lot of points are rejected due to *Shadow Effect*.

These results suggest that the simulator can be used to find an *optimum scanning zone* for the given model and scanner specification, in addition to a view plan.

## 4.2 Scanner Parameters

The simulated scanner has many configurable parameters or specifications as explained in the Section 3.3. However for the experiments, only variations in *FOV* and *resolution* are tested. Other parameters that could be tested are *baseline* and *range*. The results using different *FOV* and *resolutions* are given below:

Table 2. Effect of Standoff Distance

Model	Distance	Viewpoints	Viewplan	Cover
Mseke	50	1858	92	90.55
	100	1858	14	98.63
	300	1858	14	98.05
	500	1858	29	96.53
Humerus	5	276	17	68.05
	10	276	10	92.50
	30	276	6	95.55
	50	276	7	93.97
Mug	50	748	13	85.53
	100	748	12	89.19
	300	748	58	53.07
	500	748	107	48.88
Robinet	50	814	24	90.55
	100	814	19	92.74
	300	814	40	89.14
	500	814	34	72.71
Joueur Pétanque	100	338	9	76.79
	300	338	8	94.02
	500	338	15	86.98
St.Jean Baptiste	100	746	19	87.13
	300	746	10	86.92
	500	746	22	81.57

#### 4.2.1 Field of Vision (FOV)

The scanner FOV is one of the parameter that defines the frustum volume. The results of using different values for FOV are given in the Table 3.

It can be seen in the Table 3 that the optimum FOV is between  $60^\circ$  and  $90^\circ$ , that yields smaller view plan and a good area cover. The algorithm gives reasonable area cover for narrow FOV ( $10^\circ$ ), but length of the view plan is large. The length of the view plan is large because of narrow FOV. This results shows that the view planning algorithm is capable of producing a good area cover even for narrow FOV scanners.

#### 4.2.2 Resolution

The horizontal and vertical resolution of the scanner are the same for each experiment. The results of changing scanner resolution are given in the Table 4.

It can be seen in the Table 4 that length of the view plan increases with an increase in the scanner resolution. The percentage area cover is not significantly improved with an increase in the scanner resolution. This shows that view planning process can be efficiently performed using a scanner model with lower resolution, while actual scanning may be done at a higher resolution.

### 4.3 Positioning System Constraint

The viewpoint space is filtered by applying positioning system constraint as explained in the Section 3.4. The effect of using this constraint on viewpoint space is demonstrated using **Mseke** model. The model is successively placed at different positions relative to the base of Kuka KR16. The approximate positions are depicted on the work envelope diagram<sup>16</sup> in the Fig. 7.

The results are given in the Table 5. The first row gives result without using positioning system constraint for the **Mseke** model. The *Robot Position* is the position of the robot with respect to the center of the object. The *viewpoints* column gives number of viewpoints reachable by the positioning system as determined by solving *Inverse Kinematics*.



Table 3. Effect of Scanner FOV

Model	FOV	Viewpoints	Viewplan	Cover
Mseke	10	1858	107	97.19
	30	1858	26	97.57
	60	1858	14	98.05
	90	1858	13	97.83
Humerus	10	276	32	70.36
	30	276	9	94.94
	60	276	6	95.55
	90	276	6	94.01
Mug	10	748	58	60.43
	30	748	20	86.18
	60	748	12	89.19
	90	748	11	90.90
Robinet	10	814	66	84.61
	30	814	32	80.41
	60	814	19	87.60
	90	814	17	84.19
Joueur Pétanque	10	338	30	47.33
	30	338	11	85.03
	60	338	8	94.02
	90	338	7	90.41
St.Jean Baptiste	10	746	39	80.45
	30	746	16	88.04
	60	746	9	91.80
	90	746	9	91.65

Table 4. Effect of Scanner Resolution

Model	Resolution	Viewpoints	Viewplan	Cover
Mseke	0.5	1858	8	98.43
	1.0	1858	14	98.05
Humerus	0.1	276	5	93.16
	0.5	276	5	94.62
	1.0	276	6	95.55
Mug	0.1	748	5	98.38
	0.5	748	23	79.91
	1.0	748	58	53.07
Robinet	0.5	814	12	94.33
	1.0	814	19	87.60
Joueur Pétanque	0.5	338	5	94.93
	1.0	338	8	94.02
St.Jean Baptiste	0.3	736	8	94.02
	0.5	736	9	91.80

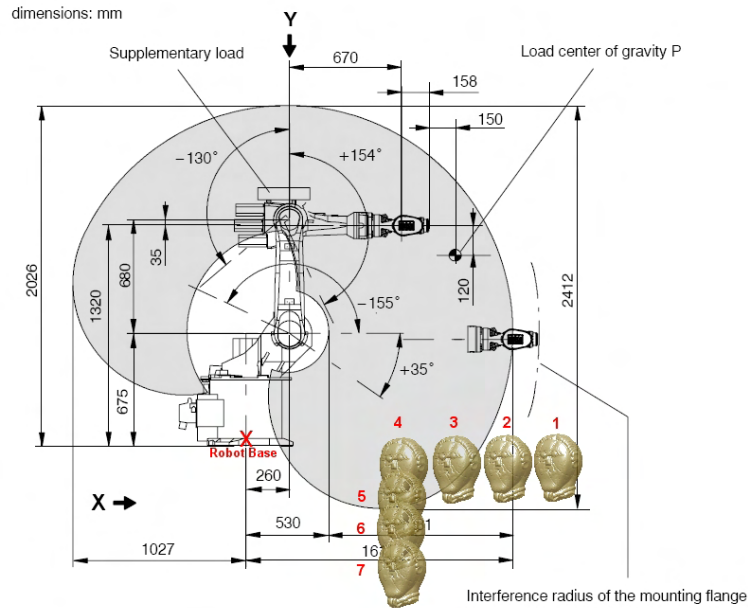


Figure 7. Approximate Positions of the Model on KR16 Work Envelope diagram

Table 5. Effect of Positioning System Constraint

No.	Robot Position	Viewpoints	Viewplan	Cover
	Mseke	930	11	97.91
1	1500, 500, 0	76	18	52.96
2	1400, 500, 0	304	23	88.28
3	1300, 500, 0	495	15	95.08
4	1200, 500, 0	692	11	97.54
5	1200, 600, 0	600	11	97.82
6	1200, 700, 0	515	15	96.43
7	1200, 800, 0	432	20	92.16

It can be seen in Table 5 that in position 1, a lot of viewpoints are rejected, because they are not reachable by the robot, as illustrated by the Fig. 7. The less viewpoints are rejected, when the object is moved towards the robot base. In position 1, when lot of viewpoints are rejected, a major part of the model is not digitized, and thus, a large hole is present in the final 3D model as shown in the Fig. 8. The holes in the model, for the positions 2 and 3 also shows that these regions were not digitized and the digitizer was not reachable due to positioning system constraints. The optimum position for the object to be placed is 4 and 5, in which entire object is digitized.

The simulator can be used to locate optimum position of the object with respect to the positioning system.

#### 4.4 Results Summary

The results of the experiments are summarized in this section. The results shows that the view planning algorithm generates optimum view plans for the various viewpoints spaces and scanner configurations. Thus the view planning algorithm is generalized and can be used with any positioning system or 3D digitizer. The results further demonstrates the ability of view planning algorithm to use a rough or coarse model and generate an optimum plan. The results suggest that the simulator can be used to find an *optimum scanning zone* for the given model and 3D digitizer specification. The optimum scanner configuration for the given model can also be found. Finally, the suitable position and orientation of the object with respect to a positioning system can be computed for real implementation.

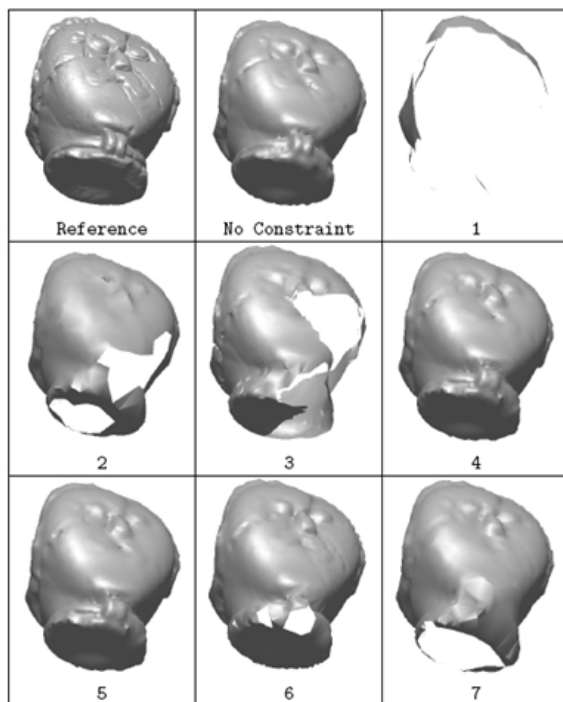


Figure 8. Visualization of Effect of Positioning System Constraint

## 5. CONCLUSION

An implementation of automatic 3D acquisition and post-processing pipeline is presented in this article. The results of the experiments shows that the view planning algorithm generates optimum view plans for various viewpoints spaces and scanner configurations. Thus the view planning algorithm is generalized and can be used with any positioning system or 3D digitizer. The experimental results further demonstrates the ability of view planning algorithm to use a rough or coarse model and generate an optimum plan.

The simulator can be used to implement a real acquisition system in many ways. It can be used to find an *optimum scanning zone* for the given model and 3D digitizer specification. The optimum scanner configuration for the given model can be found and a real scanner based on this configuration can be selected for the real system. The simulator can be used offline to generate a view plan using 3D model, or it can also be used for multistage view planning. Furthermore, the optimum location and orientation of the object with respect to a positioning system can be found.

### 5.1 Recommendations for the Future Work

The following are the recommendations for the future work:

- The model quality specification can be used in the *Measurability Matrix*, by using a visibility measure instead of a binary number.
- The solutions to *Set Covering Problem* should be further explored or the MetaRaPS<sup>20</sup> should be fine tuned.
- The views overlap constraint for the surface registration can be introduced while solving SCP.<sup>19</sup>
- The robustness of the view planning algorithm should be addressed by modeling *pose error* in the positioning system, and implementation of *pose error compensation*.<sup>12,21</sup>
- The viewpoints space can be further constrained by modeling fixtures and other objects in the environment that introduce occlusions and collision avoidance considerations.

- The optimized trajectory (or shortest path) of the positioning system should be generated from the view plan.
- The simulator should be tested in-loop with a real positioning system and a 3D scanner.

## ACKNOWLEDGMENTS

Authors would like to thank European Commission for funding this project through Erasmus Mundus Program - [www.vibot.org](http://www.vibot.org).

## REFERENCES

- [1] Scott, W. R., Roth, G., and Rivest, J.-F., "View planning for automated three-dimensional object reconstruction and inspection," *ACM Computing Surveys* **35**(1), 64–96 (2003).
- [2] K.A., T., P.K., A., and Tsai, R., "A survey of sensor planning in computer vision," **11**, 86–104 (February 1995).
- [3] Chen, S. and Li, Y., "A method of automatic sensor placement for robot vision in inspection tasks," in [*International Conference on Robotics and Automation*], 2545–2550 (May 2002).
- [4] Tarabanis, K. A., Tsai, R. Y., and Allen, P. K., "The mvp sensor planning system for robotic vision tasks," *Robotics and Automation* **11**, 72–85 (February 1995).
- [5] Abrams, S., Allen, P. K., and Tarabanis, K. A., "Computing camera viewpoints in an active robot workcell," *International Journal of Robotics Research* **18**, 267–285 (March 1999).
- [6] Abrams, S., Allen, P. K., and Tarabanis, K. A., "Dynamic sensor planning," in [*DARPA93*], 599–607 (1993).
- [7] Trucco, E., Umasuthan, M., Wallace, A. M., and Roberto, V., "Model-based planning of optimal sensor placements for inspection," *Robotics and Automation* **13**, 182–194 (April 1997).
- [8] Prieto, F., Lepage, R., Boulanger, P., and Redarce, T., "A CAD-based 3D data acquisition strategy for inspection," *Machine Vision Applications* **15**(2), 76–91 (2003).
- [9] Martins, F. A. R., García-Bermejo, J. G., Casanova, E. Z., and González, J. R. P., "Automated 3d surface scanning based on cad model," *Mechatronics* **15**, 837–857 (2005).
- [10] Pudney, C. J., *Surface Modelling and Surface Following for Robots Equipped with Range Sensors*, PhD thesis, Department of Computer Science, University of Western Australia (September 1994).
- [11] Ailisto, H., *CAD model-based planning and vision guidance for optical 3D coordinate measurement*, PhD thesis, VTT Electronics, University of Oulu (March 1996).
- [12] Scott, W. R., "Model-based view planning," tech. rep., National Research Council of Canada (March 2005).
- [13] Tarbox, G. H. and Gottschlich, S. N., "Planning for complete sensor coverage in inspection," *Computer Vision and Image Understanding* **61**(1), 84–111 (1995).
- [14] Mehdi-Souzani, C., Thiébaud, F., and Lartigue, C., "Scan planning strategy for a general digitized surface," *Computing and Information Science in Engineering* **6**, 331–339 (December 2006).
- [15] "RapidForm, <http://www.rapidform.com>."
- [16] Kuka-Robotics, "Kuka KR6, KR16, KR16 L6 specification," (2003).
- [17] Corke, P. I., "A robotics toolbox for MATLAB," *IEEE Robotics and Automation Magazine* **3**, 24–32 (March 1996).
- [18] Denavit, J. and Hartenberg, R. S., "A kinematic notation for lower-pair mechanisms based on matrices," *Journal of Applied Mechanics* , 215–221 (June 1955).
- [19] Scott, W. R., Roth, G., and Rivest, J.-F., "View planning as a set covering problem," tech. rep., National Research Council of Canada (2001).
- [20] Guanghai, L., W., D. G., and E., W. G., "An effective and simple heuristic for the set covering problem," *European Journal of Operational Research* **127**, 1387–1403 (February 2007).
- [21] Scott, W. R., Roth, G., and Rivest, J.-F., "Pose error effects on range sensing," *Vision Interface* , 331–338 (May 2002).