Development ofControl System for Virtual Manufacturing System

THESIS REPORT

Proposed as one of the requirements to Obtain a Bachelor Degree in Industrial Engineering

Created by:

Name	: Abiyoga Kristanto
Student number	: 07522110

INTERNATIONAL PROGRAM

DEPARTMENT OF INDUSTRIAL ENGINEERING

FACULTY OF INDUSTRIAL TECHNOLOGY

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2012

Declaration Letter

I declare this research was conducted by myself except several citations that have been mentioned its sources.

Yogyakarta, April 11, 2012

Abiyoga Kristanto

Student ID 07522110

THESIS APPROVAL OF SUPERVISOR

Development Control System for Virtual Manufacturing System

THESIS

By:

Name : Abiyoga Kristanto

Student Number : 07 522 110

Yogyakarta, April , 2012

Approved,

Thesis Supervisor,

(Muhammad Ridwan Andi Purnomo, ST., M.Sc., Ph.D)

THESIS APPROVAL OF EXAMINATION COMITTEE

Development Control System for Virtual Manufacturing System

Created by

Name : Abiyoga Kristanto

Student number : 07522110

Was defended in Front of Examination Committee

Proposed as one of the requirements to Obtain a Bachelor Degree in Industrial Engineering

Yogyakarta, April , 2012

Examination committee

Agus Mansur, ST., M.Eng.Sc.

Examination committee Chair

Sri Indrawati, ST., M.Eng.

Member I

Muhammad RidwanAndiPurnomo, ST., M.Sc., Ph.D.

Member II

Accepted By,

Director of International Program

Department of Industrial Engineering

Faculty of industrial Technology

Universitas Islam Indonesia

Muhammad RidwanAndiPurnomo, S.T., M.Sc., Ph.D

iv

Dedication Page

I dedicated this final project to my Dad,, my Mom,

my Sister and someone who loves me.You are my inspiration.

MOTTO

"O My Sons! Go you and enquire about Yusuf and his brother, and never give up hope of Allah's Mercy. Certainly no one despairs of Allah's Mercy, except the people who disbelieve"

(QS: Yusuf: 87)

Pat your own shoulder and says"it's not easy but I can do it, success is our right" – Mario Teguh

"He gives wisdom to anyone He wishes, and anyone He gives wisdom to, receives an abundant boon. Yet only prudent persons will reflect on it."

(QS: Al – Baqarah 269)

Acknowledgments

Assalamualaikum Wr.Wb

Praise and gratitude we pray to ALLAHSWT that has given us an abundance of grace, Taufik and his guidance. Sholawat and greetings we give to our great Prophet Muhammad SAW and his family and friends, so I can finish this thesis.

Thesis is structured as a condition for completing the study period stratum-1 at the Faculty of Industrial Technology, Islamic University of Indonesia. In completing this thesis, the author of many to get support from various parties, the authors express gratitude and appreciation as much as possible to those who have supported either directly or indirectly to:

1. Head of Industrial Technology Faculty, Ir. Gumbolo Hadi Susanto, MSc.
2. Head of Industrial Engineering Department, Drs. H. M. Ibnu Mastur, MSIE.
3. Muhammad Ridwan Andi Purnomo., ST., M.Sc., Ph.D for the supervision.
4. My parents and my Sister for giving support in the form of prayer and motivation.
5. Datamining Family, thanks for being my family.
6. For all IP FTI member, Especially Anggit, Rinda and Desta, thanks for this friendship.
7. For all member of my dermitory, Satyo, Gega, Nico, Heri, Mr.Zainal Abidin for great lifes in this four years.
8. To all party that cannot be mentioned one by one that give me supports to me in finishing the thesis.

Authors are aware there are still many shortcomings of this writing, the authors present a huge apology. Hopefully this report can be useful for writers and readers.

Wassalamualaikum Wr.Wb

Yogyakarta, April 2012

Abiyoga Kristanto

Development ofControl System for
Virtual Manufacturing System

THESIS REPORT

Proposed as one of the requirements to Obtain a Bachelor Degree in Industrial Engineering

Created by:

Name : Abiyoga Kristanto

Student number : 07522110

INTERNATIONAL PROGRAM

DEPARTMENT OF INDUSTRIAL ENGINEERING

FACULTY OF INDUSTRIAL TECHNOLOGY

UNIVERSITAS ISLAM INDONESIA

YOGYAKARTA

2012

TABLE OF CONTENT

LIST OF TABLE

LIST OF FIGURE

ABSTRACT

Recently, Virtual Manufacturing (VM) has received great attention from several researchers because of its ability to save the time and cost. In VM, analysis of manufacturing activities are carried out virtually using computer technology so that it can be finished fast. Technically, all of objects in real manufacturing system are modelled virtually in 3D space. Besides, it is need to provide a control system as well in order to move and monitor the virtual objects in 3d environment. The aim of this research is to develop a control system for a VM system. The control system is developed based on inverse kinematic concept and it is instantiated using Microsoft Visual C# technology. A commercial 3D engine, TV3D system is also utilised in order to render the modelled 3D objects. Based on testing using a case study which involves 2 industrial robot arms with 6 degree of freedom, it shows that the developed control system is able to move and monitor the industrial robot arms without any collisions. It can be said fairly that the proposed control system is work well in such case study.

Keywords: Virtual Manufacture, Control Algorithm, Inverse Kinematic, TV3D

Declaration Letter

I declare this research was conducted by myself except several citations that have been mentioned its sources.

Yogyakarta, April 11, 2012

Abiyoga Kristanto

Student ID 07522110

THESIS APPROVAL OF SUPERVISOR

Development Control System for
Virtual Manufacturing System

THESIS

By:

Name : Abiyoga Kristanto

Student Number : 07 522 110

Yogyakarta, April , 2012

Approved,

Thesis Supervisor,

(Muhammad Ridwan Andi Purnomo, ST., M.Sc., Ph.D)

THESIS APPROVAL OF EXAMINATION COMITTEE

Development Control System for Virtual Manufacturing System

Created by

Name : Abiyoga Kristanto

Student number : 07522110

Was defended in Front of Examination Committee

Proposed as one of the requirements to Obtain a Bachelor Degree in Industrial Engineering

Yogyakarta, April , 2012

Examination committee

Agus Mansur, ST., M.Eng.Sc.

Examination committee Chair

Sri Indrawati, ST., M.Eng.

Member I

Muhammad RidwanAndiPurnomo, ST., M.Sc., Ph.D.

Member II

Accepted By,

Director of International Program

Department of Industrial Engineering

Faculty of industrial Technology

Universitas Islam Indonesia

Muhammad RidwanAndiPurnomo, S.T., M.Sc., Ph.D

iv

Dedication Page

I dedicated this final project to my Dad,, my Mom,

my Sister and someone who loves me.You are my inspiration.

MOTTO

"O My Sons! Go you and enquire about Yusuf and his brother, and never give up hope of Allah's Mercy. Certainly no one despairs of Allah's Mercy, except the people who disbelieve"

(QS: Yusuf: 87)

Pat your own shoulder and says"it's not easy but I can do it, success is our right" – Mario Teguh

"He gives wisdom to anyone He wishes, and anyone He gives wisdom to, receives an abundant boon. Yet only prudent persons will reflect on it."

(QS: Al – Baqarah 269)

Acknowledgments

Assalamualaikum Wr.Wb

Praise and gratitude we pray to ALLAHSWT that has given us an abundance of grace, Taufik and his guidance. Sholawat and greetings we give to our great Prophet Muhammad SAW and his family and friends, so I can finish this thesis.

Thesis is structured as a condition for completing the study period stratum-1 at the Faculty of Industrial Technology, Islamic University of Indonesia. In completing this thesis, the author of many to get support from various parties, the authors express gratitude and appreciation as much as possible to those who have supported either directly or indirectly to:

1. Head of Industrial Technology Faculty, Ir. Gumbolo Hadi Susanto, MSc.
2. Head of Industrial Engineering Department, Drs. H. M. Ibnu Mastur, MSIE.
3. Muhammad Ridwan Andi Purnomo., ST., M.Sc., Ph.D for the supervision.
4. My parents and my Sister for giving support in the form of prayer and motivation.
5. Datamining Family, thanks for being my family.
6. For all IP FTI member, Especially Anggit, Rinda and Desta, thanks for this friendship.
7. For all member of my dermitory, Satyo, Gega, Nico, Heri, Mr.Zainal Abidin for great lifes in this four years.
8. To all party that cannot be mentioned one by one that give me supports to me in finishing the thesis.

Authors are aware there are still many shortcomings of this writing, the authors present a huge apology. Hopefully this report can be useful for writers and readers.

Wassalamualaikum Wr.Wb

Yogyakarta, April 2012

Abiyoga Kristanto

ABSTRACT

Recently, Virtual Manufacturing (VM) has received great attention from several researchers because of its ability to save the time and cost. In VM, analysis of manufacturing activities are carried out virtually using computer technology so that it can be finished fast. Technically, all of objects in real manufacturing system are modelled virtually in 3D space. Besides, it is need to provide a control system as well in order to move and monitor the virtual objects in 3d environment. The aim of this research is to develop a control system for a VM system. The control system is developed based on inverse kinematic concept and it is instantiated using Microsoft Visual C# technology. A commercial 3D engine, TV3D system is also utilised in order to render the modelled 3D objects. Based on testing using a case study which involves 2 industrial robot arms with 6 degree of freedom, it shows that the developed control system is able to move and monitor the industrial robot arms without any collisions. It can be said fairly that the proposed control system is work well in such case study.

Keywords: Virtual Manufacture, Control Algorithm, Inverse Kinematic, TV3D

TABLE OF CONTENT

CHAPTER II RESEARCH METODOLOGY

LIST OF TABLE

LIST OF FIGURE

CHAPTER I

INTRODUCTION

1.1 Background

Globalization has opened up more markets to manufacturers and generated exceeding pressure on them to provide high quality products fast, economically, and with high level of adaptability (Offodile et al., 2002), in order to create that condition, it is need a very high production cost, undoubtedly, one of the most effective cost reduction strategies is reconfiguration of the production facilities (Saadetin et al., 2010). In real manufacturing system, firms are required to make the system efficient, to create an efficient system, all aspects should be good like the layout, to create a good layout, then we should trial and error and it takes a lot of time and costly, because of its problem virtual manufacture are made. from its virtual manufacturing, we can make a real product. Recently, many companies are using Virtual Manufacturing (VM) in order to save the time and cost. VM is knowledge and information based technique developed in recent years. With the use of the VM technologies, many aspects of activities can be integrated and realized into one system. By implementing VM in manufacturing so it will give benefits on reducing cost and time.

VM is used loosely in number context. It refers broadly to the modeling of manufacture system and components with effective use of audiovisual and other sensory features to simulate or design alternative for an actual manufacturing environment, mainly through effective use of computers (Prashant et al., 2001). There is an increasing needs to generate detailed real time virtual environments that closely mimic real world facilities (Fumarola et al., 2011). The motivation is to enhance the ability to predict potential problems and inefficiencies in product functionality and manufacturability before real manufacturing occurs.

To create a VM use 4 types of models, there are the virtual device model (object model), the transfer handler model (functional model), the state manager model, and the flow controller model (dynamic model). The object model describing the objects in the system, and their relationships. The dynamic model describing the interactions among objects in the system. The functional model describing the data transformations of the system. The state manager model is use to maintaining decision variables based on the mapping relations between decision variables and the states of virtual devices in the system (Park., 2005). This research will construct the one of VM and create the algorithm to control the model.

Previous research presented a technique to establish VM. Bruno et al,. (2000) discuss about controlling the interaction of multilink flexible arm in contact with a compliant surface. For a given position and surface stiffness, the joint and deflection variables are computed using a closed loop inverse kinematics algorithm.

The problems that emerged in this research is to develop a specific VM with control algorithm. It contains an introduction to virtual manufacturing, divided into a general part on modeling and simulation, verification, validation and acceptance. VM must be developed because of there are some issues like the manufacturing cost and time to market can be reduced.

In making the VM there are several stages, the first stage is the selection of a model or models of development, the second stage is to develop control algorithms, the third stage is to scheduling, finished product, and inventory, the fourth stage is the estimated cost. In the present study will focus on the second stage of develop control algorithms.

1.2 Problem Formulation

Based on the background to the study, hence it can be defined the problem formulation is:

1. How to develop and control system to control VM models in virtual environment?

1.3 Problem Boundary

Bounding the problem need to be defined in order to focus the research so that the research objective can be well-achieved. So, the problem boundary can be defined as follow:

1. The virtual manufacturing system is not discussed about finish product.
2. The virtual manufacturing system is not discussed about inventory.
3. The virtual manufacturing system is not discussed about the cost.

1.4 Research Purpose

Research purpose of the research is:

1. To develop virtual Manufacture.
2. To develop and control system to control VM models in virtual environment.

1.5 Benefit of Research

This research can give its benefit to:

1. To enhance the knowledge of manufacturing system.
2. To enhance the knowledge about control system.
3. To enhance control algorithm technique for the object.

1.6 Writing Systematic

This thesis report will be arranged to some chapter, and every chapter will be explained below:

CHAPTER II LITERATURE REVIEW

Contains both the concept and basic principles needed to solve research problems. Besides it also includes a description of the results of studies that have been done before by other researchers who have anything to do with the research undertaken.

CHAPTER III RESEARCH METHODOLOGY

This chapter includes the research object, method of collecting data, kind and source of the data, tools that use flow diagram of the research, procedure, and data analysis.

CHAPTER IV COLLECTING AND PROCESSING DATA

In this section contains the data obtained during the study and how to analyze the data. Processing result is displayed in tables and graphs. The definition of data processing also includes the analysis conducted on the results obtained. In this section is a reference to the discussion of the results of which will be written in section V of the discussion of the results.

CHAPTER V DISCUSSION

Discussing the results obtained in research and compliance with the objectives of research so as to produce a recommendation.

CHAPTER VI CONCLUSION AND SUGGESTION

Contains the conclusions of the analysis and recommendation or suggestions which achieved and problems found during the study, so needs a recommendation to be studied in further research.

CHAPTER II

LITERATURE REVIEW

2.1 Previous Research

At this time, VM is the common issues that experiencing development. There are many research of VM. Some of them are Park (2005) describe about A methodology for creating a virtual model for a flexible manufacturing system. This paper presents an object-oriented methodology for the modeling and simulation of a virtual FMS. For the implementation of the proposed virtual FMS model, the paper employs DEVSIM++ which is an object-oriented simulation language based on Zeigler's DEVS formalism. The OOM paradigm of Rumbaugh uses three kinds of models to describe a system: the object model, describing the objects in the system and their relationships; the dynamic model, describing the interactions among objects in the system; and the functional model, describing the data transformations of the system. The proposed virtual FMS model follows the OOM paradigm and uses four types of components: the virtual device model corresponding to the object model, the transfer handler model corresponding to the functional model, and the state manager and flow controller model corresponding to the dynamic model. A virtual device is designed considering reusability, and the device consists of two parts: shell and core the shell part allows a virtual device model to adapt to different FMS configurations, and it encloses the core part which contains the inherent properties of a device, such as kinematics, geometric shape, and the execution of device-level commands. For the fidelity of the virtual FMS model, a transfer handler model has a set of device-level commands imitating the physical mechanism of a transfer. As a result, we can check the mechanical validity of each transfer and also expect more accurate simulation results, because the total time of each transfer will be computed from its physical mechanism. The flow controller model makes decisions on fir able transfers based on decision variables, which are maintained by the state manager model. To maintain the decision variables, the state manager stores mapping relations between decision variables and the states of virtual devices in the system. The mapping relations can serves as the guidelines for the planning of a real FMS implementation.

Saadetin et al., (2010) describes about a genetic algorithm (GA) based heuristic approach for job scheduling in virtual manufacturing cells (VMCs). In this paper, their investigated the scheduling of virtual manufacturing cells (VMCs), where a virtual cell (VC) is a group of machines temporarily assigned to processing a job, and the machines are not necessarily in physical proximity. Machines that have the similar processing abilities are located different areas in the shop floor to enhance the system's agility. Separation of machines in shop floor results in increased total traveling distance. So, first develop a multi-objective mixed integer programming formulation to characterize the scheduling problem. The objectives, makes pan and total traveling distance, are converted a single objective by allocating them weights. The complex nature of the problem enforces us to develop a heuristic approach to find good solutions. For this aim, researcher develops a genetic algorithm approach. (Ga et al.'s) two-vector representation scheme is used and an effective decoding method interprets each chromosome into an active schedule. A reorder procedure is used to unify the operation sequence in the chromosome with the sequence in the decoded schedule to facility genetic operators to pass on the good traits to their offspring. In order to improve the heritability of crossover operation, operation sequence of (Gen et al.'s) representation is changed to the format of permutation representation. To measure the effectiveness of GA, we generate 45 test problems and compare the MIP solutions with GA solutions. For the 20 problems, MIP model find the optimal solution and GA find the same solutions. Sixteen problems out of the rest 25 problems, GA outperforms MIP model in terms of solution quality. GA is also promising and efficient in terms of solution time. So, GA can easily be substituted in the place of MIP model.

Slomp et al., (2005) describe about the main objective of the paper is to illustrate a new operational procedure of creating virtual manufacturing cells periodically, for instance every week, to meet the demand turbulence in markets. We define a virtual manufacturing cell as a temporary grouping of machines, jobs and workers. The grouping is meant to focus machines and the minds of workers to families of jobs. This supports the minimization of setup time losses. It also promotes the novel idea of team production. This paper shows the applicability of mathematical programming models to create virtual manufacturing cells. We assume in our approach that small cells, in terms of the number of required FTEs, simplify the realization of sequentially performing jobs of the same family on the same machine. We first present a general formulation for the design of virtual cells. In order to solve the problem for a realistic problem size, the general integer programming formulation is decomposed in two stages, which can be solved sequentially. The models in the two stages can be applied to maximize the setup savings, to minimize the number of inter-cell movements, and to keep the machine coverage and multi-functionality on a sufficient level. The goal programming models in both stages facilitate rapid reformulation in case of changes in parameter values such as families, cells, etc. Most changes can easily be incorporated in the models by simple modifications in the LINGO formulation. An illustrative example is given to indicate the usefulness of the models in the two stages.

Xue Yan and P Gu., (1996) describe about the important of product features, quality, cost and time to market for a manufacturer to remain competitive. Rapid prototyping systems offer the opportunities to make products faster and usually at lower costs than using conventional methods. Since RP&M can substantially reduce the product development cycle time, more and more businesses are taking advantage of the speed at which product design generated by computers can be converted into accurate models that can be held, viewed, studied, tested, and compared. Several new and promising rapid prototyping manufacturing techniques were discussed. They are all based on material deposition layer by layer. Each of them has particular features in terms of accuracy, material variety and the cost of the machine. Some present problems and research issues were also discussed. This is a rapid development area. Capacities and the potential of rapid prototyping technologies have attracted a wide range of industries to invest in these technologies. It is expected that greater effort is needed for research and development of those technologies so that they will be widely used in product oriented manufacturing industries.

Fok et al., (2009) describe about an intelligent virtual environment to support cognitive rehabilitation. The cognitive rehabilitation process is not only time consuming and labor intensive, but also can consume enormous training and financial resources. With the advancement of computer technology, intelligent virtual environments could become useful tools for future cognitive assessment and rehabilitation. This paper proposes a framework for an intelligent virtual cognitive rehabilitation environment and presents the development of a prototype system to guide patients to relearn meal preparation skills. Based on existing conventional rehabilitation activities, a virtual kitchen suitable for use with various with electronic exercises was developed. This virtual kitchen allows the patient to learn and practice proper meal preparation skills with guidance from an intelligent assistant. The intelligent assistant uses fuzzy logic to output appropriate visual and audio cues based on the input patient's behaviors. The results of a preliminary clinical investigation of the intelligent virtual environment for a simple cooking task are promising. To fully realize the potential of the intelligent virtual environment for cognitive rehabilitation, the prototype system has to be further improved. With advancement in artificial intelligence and data-mining, the concept of an intelligent virtual therapist residing in the virtual environment could be the next goal. This could result in rehabilitation activities that can adapt to the patient's conditions and performances and a system with greater sophistication and responsiveness. At this stage, much work remains to be done. Nevertheless, it is hoped that the deliberation, framework and lessons learned arising from this work would be useful for the advancement of interactive computer-based therapeutic tools for people with cognitive problems.

Amos et al., (2008) describe focuses on the use of virtual manufacturing for press line monitoring and diagnostics. A machine service support system (MSSS) has been designed and implemented. The system is an example of advanced use of 3-D graphical simulation tools in the resource domain. It extends the use of simulation models from the machine system design and development phase into the operational phase. This, however, requires special attention for the synchronization of the simulation and the various signals. MSSS can be used for continuous on-line monitoring or for analysis of previous operation with the use of media player functions. In the latter case, the user cans remotely configure the data logging to avoid logging of redundant data. Whilst MSSS may be beneficial to almost any system integrator or machine builder installing equipment at remote locations, suppliers of press lines and press cells can benefit in particular as a problem with any device in such lines or cells may hamper the operation seriously and not seldom result in a complete production stop. As a result, after successfully testing MSSS in a fully industrial pilot implementation, one of the project partners is considering to make MSSS available to their customers as an add on for their press lines and cells. MSSS in a broader context is an example of a synthetic environment that is used to create an enhanced ‘‘situation awareness’’ for human decision support in an IF context. Lack of this research is there are no more advanced functions for diagnostics and fault detection algorithms that may allow some decisions to be taken by the system automatically but also will support the human decision maker with better information in general. in short, the researchers did not use a control algorithm to reduce the fault detection algorithm.

Bruno And Luigi (2000) deals with the problem of controlling the interaction of a multilink flexible arm in contact with a compliant surface. The first stage is in charge of solving the inverse kinematics problem to compute the desired vectors of the joint and the de#ection variables that place the flexible arm tip at the desired position with the desired contact force. The solution is based on the transpose of a suitably modifed arm Jacobian so as to account for the static e!ects due to gravity and contact force. The computed variables are used as the set-points for the second stage, which is a simple PD joint regulator. The attractive feature of the scheme is that it does not require force and deflection measurements. The price to pay is that an exact knowledge of the arm kinematic model, link stifness and mass distribution as well as environment sti!ness and position is required to guarantee accurate regulation of tip position and contact force.

After seeing the presentation of the above studies it can be concluded that research on the development of control algorithms for the VM has not been done so that this study can be considered as new research is expected to contribute to both the world's manufacturing and education.

2.2 Basic Theory

2.2.1 Virtual Manufacturing

Virtual manufacturing is used loosely in number context. It refers broadly to the modeling of manufacture system and components with effective use of audiovisual and other sensory features to simulate or design alternative for an actual manufacturing environment, mainly through effective use of computers. The motivation is to enhance our ability to predict potential problems and inefficiencies in product functionality and manufacturability before real manufacturing occurs (Prashant et al., 2001). Virtual manufacturing system is organized from component of machine to collection of machines in a cell to multiple cells on a shop floor and extending to all other processes on factory floor.

In a real manufacturing system organization, the machine or processes are rarely moved from one place to another. On the other hand, the virtual manufacturing system has inherited characteristics of relocation during planning in order to explore optimal output under long term schedule. For short term schedules, while the machines remain at the designated position, the optimal part routing can be explored. This part routing maybe within factory premise or beyond (Khan et al., 2011).

Virtual manufacturing is the use of information technology and computer simulation to model real world manufacturing processes for the purpose of analyzing and understanding them. Unlike in the classical discrete event simulation, an easily understood but complex three-dimensional animation models are used to engineer the real manufacturing environment. Machines, machine cells, parts, and facilities can be designed and evaluated onscreen before actual facilities or products are constructed. In some instances the actual simulation could be carried on concurrently as the manufacturing facility is being built. The advantages of such an approach are many. It enables the manufacturer to speed up the time to market by integrating product development and production so that the system and parts are tested out in real time on a computer while at the same time allowing for ‘‘what if’’ type scenarios and conservation of valuable capital to take place. As Daly puts it, the ‘‘simulation tools are so powerful that designers can produce a perfect product on the first try without any scrap and without building a prototype’’. This level of agility in manufacturing has been made possible by advancements in information technology and its ubiquity fueled by the Internet and electronic commerce (e-commerce) (Offodile et al., 2002).

A. The benefit of virtual manufacture

Lee et al., (2001) talk about the benefit derived from virtual manufacture are as follow:

1. Enhancing the capability of risk measures and control. Virtual manufacture can be used to predict the cost of product development and production as well as provide the information related to the production process and the process capability. The information is useful for improving the accuracy of the decisions made by the designer and the management. The problems in product development and manufacturing processes can also be predicted and resolved prior to the actual production.
2. Shrinking the product development cycle. Virtual manufacturing will allow more computer based product model to be developed and prototyped upstream in the product development process. This will reduce the need for the number of downstream physical prototypes traditionally made to validate the product models and new design. Thus, the company can reduce its product development time.

3. Enhancing the competitive edge of an enterprise in the market. Virtual manufacture can reduce the cycle time and cost in product development. With the virtual environment provide by virtual manufacture, the customer can take part in the product development process. The design engineers can responds more quickly to the customer queries and hence provide the optimal solution to the customers. The competitive edge of an enterprise in the market can thus be enhanced.

B. The application of virtual manufacture.

Lee et al., (2001) talk about some typical applications of virtual manufacture are as follows:

1. Virtual manufacture can be used in the evaluation of the feasibility of a product design, validation of production plan, and optimization of the product design and processes. These reduce the cost in product life cycle.
2. Virtual manufacture can be used to test and validate the accuracy of the product and process design. For example, the outlook of product design, dynamic characteristic analysis, checking for the tool path during machining process, NC program validation, is checking for the collision problems in machining and assembly.
3. With the use of virtual manufacture on the internet, it is possible to conduct training under a distributed virtual environment for the operators, technicians and management people on the use of manufacturing facilities. The cost of training and production can thus be reduced.

4. As a knowledge acquisition vehicle, virtual manufacturing can be used to acquire continuously the manufacturing know-how, traditional manufacturing processes, production data, etc. This can help to upgrade the level of intelligence of a manufacturing system.

2.2.2 Virtual Environment

Virtual environment system differ from previously developed computer centered system in the extent to which real time interaction is facilitated and in terms of several characteristic, that the perceive visual space is three dimensional rather than two dimensional, the human machine interface is multimodal and the operator is immersed in computer generated environment.

Virtual environment, a commonly used definition of virtual environment is an interactive, virtual image display enhanced by special processing and by non-visual display modalities, such as auditory and haptic, to convince user that they are immersed in a synthetic space. The term immersion refers to the fact that the user gets the feeling that he or she is immersed in the computer environment, the screen separating the user and the computer appears nonexistence to the user (Prashant et al., 2001).

A The Nature of Immersive Environment

According to Wang et al., (2011) virtual reality is a technology that simulates object and spaces through 3D computer generated models. In a virtual reality model, the feeling of realism is derived from a sequence of high resolution, stereoscopic images. If the display allows viewers to project themselves into the scene, then a virtual environment is created. If the scene is shown in full scale and viewers are surrounded by 3D images, an immersive environment is generated. In the immersive environment of c2/c4 (both are three side) and c6 (a six-sided) at Lowa State University, which are full scale with high resolution; users have the sense of “being there”, or the sense of presence experienced in the environment resulting from cognitive processes.

The sense of presence is generated from human sense of sight, sound, taste, smell, and touch. In virtual environment, three conditions are required to generate the sense of presence through perception (Lombard and Ditton., 1997) image quality, image dimensions and view distance.

According to Wang et al., (2011) the immersive projection system fulfills these requirements, for it closely approximates actual size and distance with full scale, high resolution 3D objects generated in real time. For instance, C2 and C4 both are 12’ by 12’ space, in which the user is surrounded by three dimensional images, projected in real time on three walls and the floor (currently, C2 is replaced by C4). C6 is a 10’X10’X10’ room in which computer generated images are projected on all four walls, the ceiling, and the floor to deliver an enclosed, interactive, and fully immersive experience. Along with the high end projectors, the system can produce up to 4096X4096 pixel images totaling over 16.7 million pixels per wall. Forty eight dual CPU workstations send images to 24 digital cinema projectors. Images generated by these projectors would have high resolution of approximately 1165 pixels per square inch. This resolution gives users a clear and detailed display of virtual environment.

B. Design in Virtual Environments

A virtual architectural Design Tool (VADeT) was created in the C2, which had a number of metaphorical icons that served as design tools for generating, modifying and editing three dimensional objects of architectural elements. There were also tools for defining dimensions, materials, and colors of the objects. By using these tools in the system, users could create a design in a synthetic virtual reality facility (Cru-Neira et al., 1993; Chan et al., 1999)

2.2.3 Basic Concepts in Robotic

Industrial robots are beginning now to revolutionize industry. These robots do not look or behave like human being, but they do the work of human. Robots are particularly useful in a wide variety of industrial application, such as material handling, painting, welding, inspection, and assmbly. Even more impressive, however, is the new perspective thats robots may bring to the factory of the future.

Industrial robots, as other modern manufacturing systems, are advanced automation systems that utilize computer as integral parts of their control.Computers are now a vital part of industrial automation. They run production lines and control stand alone manufacturing system, such as various machine tools, welders, inspection system, and laser-beam cutters. Even more sophisticated are the new robots that perform various operations in industrial plants and participate in full automation of factories (koren, 1985).

A. Basic Structure of Robots

The industrial robot is a programable mechanical manipulator. Capable of moving along several directions, equipped at its end with a work device called the end effector and capable of performing factory work ordinarily done by human beings. The term robot is used for a manipulator that has a built-in control system and is capable of stand alone operation.

One popular dictionary define a robot as “any mechanical device operated automatically to perform in a seemingly human way.” By this definition, a garage door opener, which automatically opend the door by remote control, could also be a robot. Obviously this not an industrial robot. The Robotics International of the Society of Manufacturing Engineers (RI/SME) defines the industrial robot as “a reprogramable multi fuctional manipulator design to move material, part, tools or specialized devices through variable programmed motions for the performance of variety of task. However, the RI/SME definition of an indsutrial robot should include also the following key words: motion along several directions (degree of freedom), end effector, and factory work. Modern robotic system consist of at least three major parts, which are:

a. The manipulator, which is mechanical moving structure.
b. The drives to actuate the joints of the manipulator.
c. The computer as a controller and storer of task programs.

In general, the structure of a robot manipulator is composed of a main frame and a wrist with a tool at its end. The tool can be a welding head, a spray gun, a machining tool, or a gripper containing open-shut jaws, depending upon the specific application of the robot (Koren, 1985).

B. Robot Anatomy

Robot anatomy is concerned with the physical construction of the body, arm, and wrist of the machine. Most robots used in plants today are mounted on a base which is fastened to the floor. The body is attached to the base and the arm assembly is attached to the body. At the end of the arm is the wrist. The wrist consist of a number of components that allow it to be oriented in a variety of positions. Relative movements between the various components of the body, arm and wrist are provided by a series of joints. These joints movements usually involve either rotating or sliding motions. The body, arm, wrist assembly is sometimes called manipulator.

Attached to the robot's wrist is a hand. The technical name for the hand is "end effector". The end effector is not considerd as part of robot's anatomy. The arm and body joints of the manipulator are used to position the end effector, and the wrist joints of the manipulator are used to orient the end effector (Koren, 1985).

C. End Effectors

For industrial applications, the capabilities of the basic robot must be augmented by means of additional devices. We might refer to these devices as the robot's peripherals. They include the tooling which attaches to the robot's wrist and the sensor systems which allow the robot to interact with its environment.

In robotics, the term end effector is used to describe to the hand or tool that is attached to the wrist. The end effectorrepresent the special tooling that permits the general purpose robot to perform a particular application. This special special tooling must usually be designed specifically for the application.

End effectors can be devided inti two categories: grippers and tools. Grippers would be utilize to grasp an object, usually the work part, and hold it during the robot work cycle. There are variety of holding methods that can be used in addition to the obvious mechanical means of grasping the part between two or more fingers. These additional methods include the use of suction cups, magnets, hooks, and scoops. A tool would be use as an end effector in applications where the robot is required to perform some operation on the workpart. These application include spot welding, arc welding, spray painting, and drilling. In each case, the particular tool is attached to the robot's wrist to accomplish the application (Koren, 1985).

D. Robot Arm Kinematics and Dynamics

Robot arm kinematis deals with the analytical study of the geometry of motion of a robot arm with respect to a fixed reference cordinate system without regard to the force /moments that cause the motion. Thus kinematics deals with the analytical description or the spatial displacement of the robot as the function of time, in particular the relation between the joint variable space and the position and orientation of the end-effector of a robot arm.

There are two fundamental problems in robot arm kinematics. The first problems is usually referred to as the direct(or forward) kinematics problem, while the second problem is the inverse kinematics (or arm solution) problem. Since the independent variable in robot arm are the joint variables, and a task is usually stated in terms of the reference coordinate frame, the inverse kinematics problem is used more frequently. Denavit and Hartenberg (1955) proposed a systematic and generalized approach of utilizing matrix algebra to describe and represent the spatial geometry of the link of a robot arm with respect to a fixed reference frame. This method uses a 4 x 4 homogeneous transformation matrix to describe the spatial relationship between two adjacent rigid mechanical links and reduce the direct kinematics problem to finding an equivalent 4 x 4 homogeneous transformation matrix that relates the spatial displacement of the hand coordinate frame to the refence coordinate frame. These homogeneous transformation matrices are also useful in deriving the dynamic equations of motions or a robot arm. In general, the inverse kinematics problem can be solve by several techniques. The most commonly used methods are the matrix algebraic iterative, or geometric approach (Fu et al, 1988).

E. Robot Programming Languages

One major obstacle in using manipulators as general purpose assembly machines is the lack of suitable and efficient communication between user and the robotic system so taht the user can direct the manipulator to accomplish a given task. There are several ways to coomunicate with a robot, and the three major approaches to achieve it are discreate word recognition, teach and playback, and high level programing languages.

Current state of the art speech recognition is quite primitive and generally speaker dependent. It can recognize a set of discrete words from a limited vocabulary and usually requires the user to pause between words. Although it is now possible to recognize words in real time due to faster computer components and efficient processing algorithms, the usefulness of discrete word recognitionto describe a task is limited. Moreover, it requires a large memory space to store speech data, and it usually requires a trining period to build up speech tamplate for recognition.

The method of teach and playback involvesteaching the robot by leading it through the motion to be performed. This is usually accomplished in the following steps: (1) leading the robot in slow motion using manual control through the entire assembly task, with the joint angles of the robot at appropriate locations being recorded in order to reply the motion; (2) editing and playing back the taught motion; and (3) if the taught motion is correct, then the robot is run at an appropriate speed in repetitive motion. This method is also known as guiding and is the most commonly used approach in present day industrial robots (Fu et al, 1988).

F. Robot Intelligence

A basic problem in robotics is planning motion to solve some prespecified task, and then controlling the robot as it executes the command necessary to achieve those actions. Here, planning means deciding on a course of action before acting. This action synthesis part of the robot problem can be solved by a problem solving system that will achieve some stated goal, given by initial situation. A plan is, thus, a representation of a course of action for achieving a stated goal.

Research on robot problem solving has led to many ideas about problem solving system in artificial intelligence. In typical formulation of robot problem we have robot that is equipped with sensors and a set of primitive action that it can perform in some easy to understand world. Robot actions change one state, or configuration, of the world into another. In the “block world,” for example, we imagine a world of several labeled block resting on a table or each other and a robot consisting of a tv camera and a moveable arm and hand, that is able to pick up and move blocks. In some situations the robot is a mobile vehicles with a tv camera that performs task such as pushing object from place to place in an environment containing other objects (Fu et al, 1988).

2.2.4 Truvision 3D

The TV3D SDK is a complete 3D middleware solution for programmers looking to create anything from next generation games to complex simulations. By using our complete API and favorite development language, it can write less code, and get more done in a shorter amount of time. The entire TV3D SDK has been built from the ground up to give the programmer total control over every aspect of their 3D world.

Complete control is maintained through a very easy to learn system of objects, each with a very specific set of functions. This easy to learn system still gives all the power of programming with DirectX, without having to learn a complex API. Truevision3D is focused on building cutting edge development tools at an affordable price. Large studios, indie developers, universities, and government agencies across the globe use our products for games, simulations, training, broadcast, and more.

2.3.4 Research Comparative

From the table below, can be compared to studies conducted by previous studies that note the difference.

Table 2.1 Research Comparative

Title	**Researcher**	**Using 6 Degree of freedom**	**Using Control Algorithm**	**Using VM simulation**
A Novel Robust Algorithm robotic manipulator	Ronald A. Perez & James C. Brendelson	✗	✓	✗
Inverse kinematics in robotics using neural networks	Sreenivas Tejomurtula & Subhash Kak	✓	✓	✗
Development of Control System to Control Virtual Model	Abiyoga Kristanto	✓	✓	✓

CHAPTER III

RESEARCH METHODOLOGY

This chapter contains an explanation about the object of research and development of models to illustrate the system in preparation for planning. Stages of the research will also be described in the flow chart. All sub chapters will be explained as follows:

3.1 Research Object

Object of research in this study is a virtual factory in the form of a virtual conveyor used for assembly lines, virtual robot arm with 6 degrees of freedom, a virtual NC drilling, and a virtual box.

3.2 Development System

Several stages to make virtual manufacturing is:

a. Developing Virtual Model

This is an early stage in the development of virtual manufacturing, virtual model of conveyor, a virtual model of the robot arm, the virtual rack models and virtual model of Ncdrilling..

b. Developing Virtual Environment

Making virtual environment in purpose as the space placement for virtual models that have been built. Making virtual environments using 3D Truevision software. with the environment it will look more attractive.

c. Developing Control Algorithm

Making this algorithm aims to combine virtual models that have been made earlier to the virtual environment. The merger is intended for the virtual model can be run and can be controlling in software.

3.3 Requirement Data

The requirement data to conduct this research is:

a. The data model details.
b. Position data of each object.
c. Transfer data of moving object.
d. Rotation data of moving object.

The data obtained from previous research, so in this study does not discuss how to obtain these data.

3.4 Data Analysis

The data has been collected during the study and then analyzed so that it becomes meaningful. The process of data analysis going on since the data collected until the end of the study with the direction of research questions that are equipped with the data obtained..

3.5 Tool Analysis

In this study, researchers used several tools to complete the study. Some tools that are Truevision and Visual Studio. Truevision is useful for creating a virtual 3d model commonly used to build a gaming application. While Visual Studio is a programming language used to build a software.

3.6 Research flow diagram

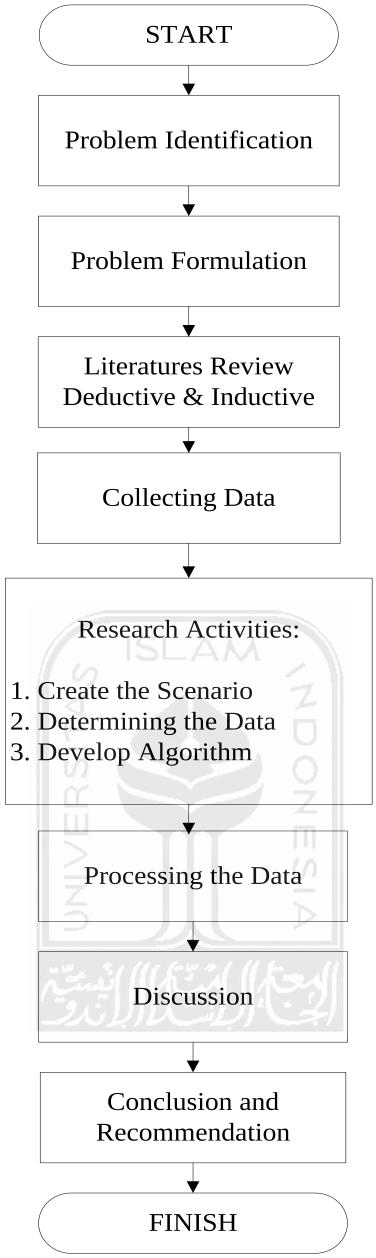


Figure 3.1 Research Flowchart

3.7 Problem Identification

Problem identification is identifying the problem that appear in making virtual manufacturing which is focused on preparation of schedulling in production system. This process is done by direct observation.

3.8 Problem Formulation

Explaining about the critical issues that arise in creating a virtual manufacturing model and the analysis that causes the problem.

3.9 Literature Review

Discusses related research associated with this research. In this study focused more on virtual manufacturing, virtual environments, rapid prototyping, and robotics.

3.10 Research Architecture

2.10.1 Create Scenario

Create a scenario of early stages in the manufacture of a VM. This section describes the VM development planning.

2.10.2 Determining the Data

Determining the data is the process whereby data generated in previous research will be in the process and offered guidance in the application of inverse kinematics.

2.10.3 Develop Algorithm

This is the core stage in the research that is making the algorithm for controlling the robot movement and the movement of moving objects from one coordinate to another coordinate.

3.11 Processing the Data

Data processing is performed using data from previous research conducted by Afrilia DR Then the data it is enforced into the functions contained in the Truevision 3D.

3.12 Discussing

This section discusses the results of modeling that has been built which will then be the basis for decision making and recommendations for further research.

3.13 Conclusion and Recomendation

This stage is the final step in this research which contains a summary of research and advanced suggestions for further research that is useful to develop research and provide recommendations necessary to complete the study.

CHAPTER IV

SYSTEM DEVELOPMENT

4.1 Research Activities

4.1.1 Research Scenario

In conducting the study required a scenario, the scenario is a sequence of events that will be done in a VM system. This scenario is intended to make research easier in designing the movement of any model used in the VM system. Robot movement is facing a raw table, then take the raw material at the starting point of 0,20,80, the starting point is located on the storage 1. Then the raw material stuck to the end of the robot 1 and then aim and release the raw material at a second point on the conveyor 1, which is at the point 60, 70.80. Then the raw material moves over the conveyor 1 to point 220,70,80 which will be taken by the robot 2. Then the robot 2 will be facing the conveyor 1 and take the raw material. raw material stuck to the end of the robot 2, robot 2 rotates to a point 300, 107, 110 in nc drilling and then release the raw material. Then Nc drilling will process the raw material to be finished product. robot 2 takes the finish product in nc drilling and finished product itself will be attached at the end of the robot 2, robot 2 rotates to a point 220, 70, -70 on the conveyor 2 and release the finished product at that point. Finish product will move on the conveyor 2 to point 60, 70, -70, which will be taken by the robot 1 and put the finished product into storage 2. More detail can be seen in Figure 4.1.

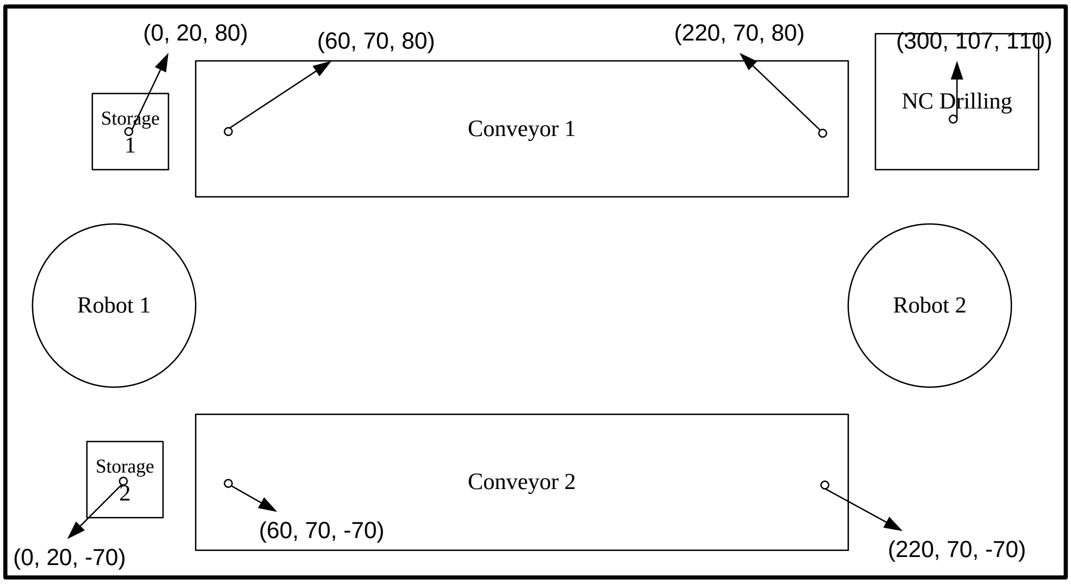


Figure 4.1 Layout Scenario

4.1.2 Determining the Data

In the development of VM takes several objects, including a robot consisting of a waist, shoulder, upper arm, fore arm, arm roll, pitch hand, hand roll, and tools which will then be assembled into a KUKA robot 15. Then using a conveyor, ncdrilling, and box. Waist has a function as the support of all part robot arm, shoulder is the second part of the robot arm that can rotate in a circle or twisting according to the magnitude of the desired degree, upper arm can rotate in a rolling or revolving in accordance with the amount of the desired degree, fore arm can move in a rotational fit with the amount of the desired degree, roll arm can rotate in a cirlce or twisting in accordance with the amount of the desired angle, pitch hand can rotate in a circle or twisting in accordance with the amount of the desired angle, roll hand can move in a twisting according to the magnitude of the desired angle, the tool is an additional section serves as a pick up and drop objects to be moved. Parts of the robot can be seen in Figure 4.2.

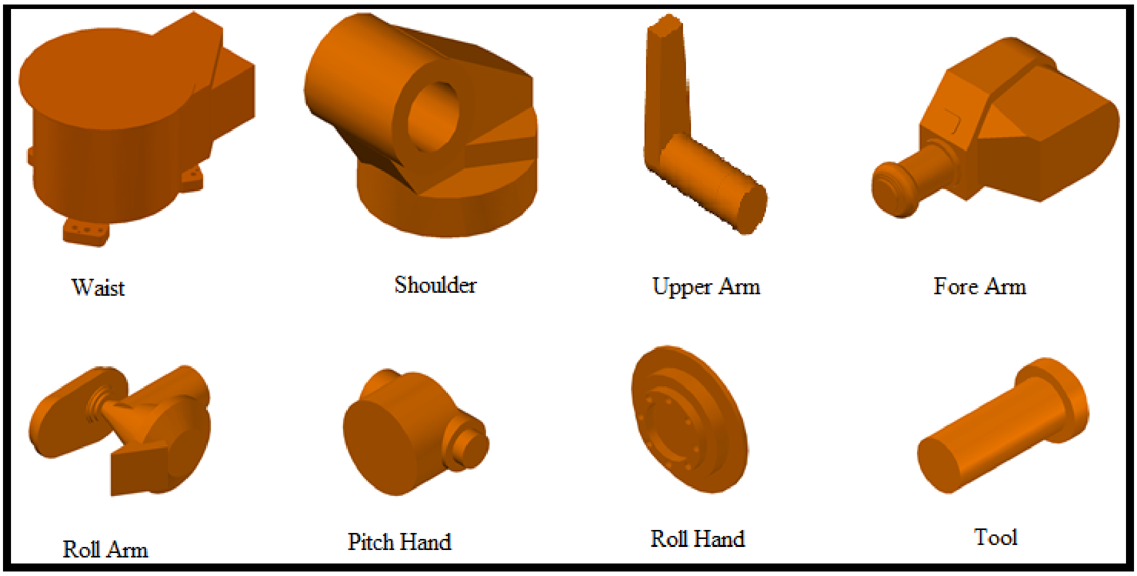


Figure 4.2 Part of Robot Arm

Parts of the robot arm above will be compiled into a KUKA robot arm 15. KUKA Robot Arm 15 is shown in Figure 4.3

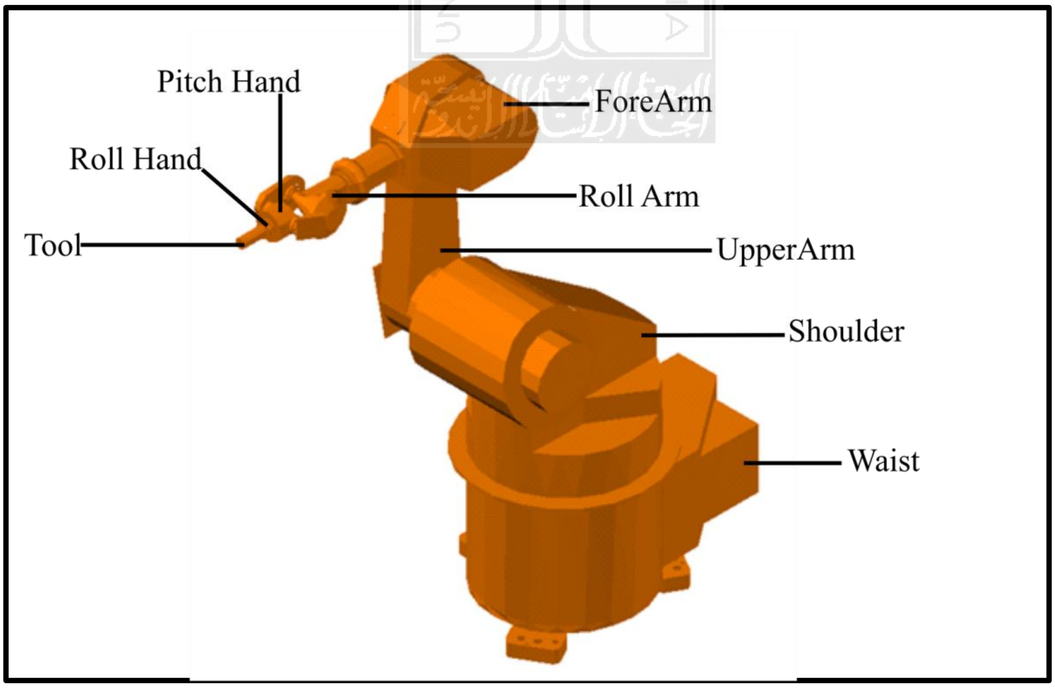
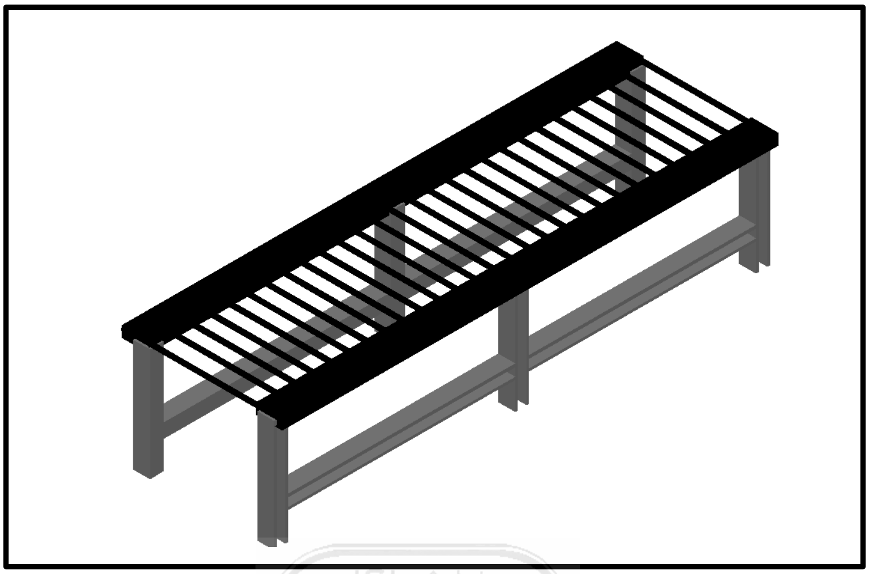
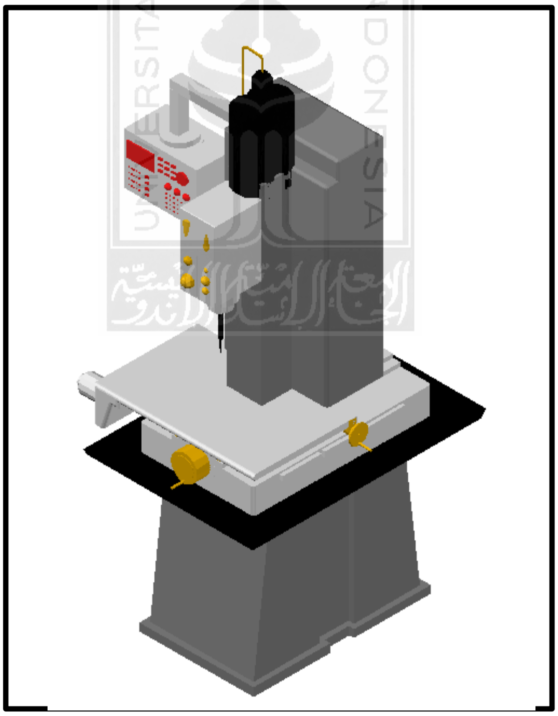


Figure 4.3 Robot Arm KUKA 15

In addition to the robot arm, researchers used a virtual conveyor, ncdrilling virtual and virtual box. Conveyor, ncdrilling and the box can be seen in Figure 4.4; 4.5; and 4.6.



Firgure 4.4 Virtual conveyor



Firgure 4.5 Virtual NCDrilling

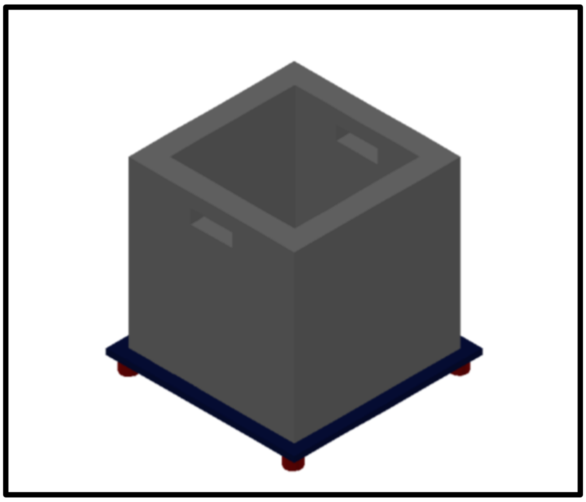


Figure 4.6 Virtual Box

All of the sections above will be prepared in accordance with a layout that has been set previously. Layout of parts above can be seen in Figure 4.7.

Figure 4.7 Layout of VM parts

4.1.3 Develop Control system

The purpose of building a control system is to control the movement of the VM model according to the scenario. Control system are used specifically on the robot arm. This research choosing inverse kinematic to move the robot arm. Inverse Kinematic Given an orientation and position for a robotic arm, it want to show That by finding all possible combinations of joint settings, it can place the hand of the robot at this exact point and orientation. This research choosing the method because the inverse kinematic destination point of the robot is known in advance. To apply the method, researcher need a basic matrix for each part of the robot arm to move the robot. Matrixs is obtained from studies previously. This matrix can be seen in Table 4.1.

Table 4.1 Basic Matrix for Each Parts of Robot

Robot 1	**Robot 2**
waist--------------------------- -000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 -000.10 0000.00 0010.00 0000.00 0010.00 0001.00	waist2--------------------------- 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0280.00 0000.00 0010.00 0001.00
shoulder------------------------- -000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 -000.10 0000.00 0010.00 0042.10 0010.00 0001.00	shoulder2------------------------- 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0280.00 0042.10 0010.00 0001.00
upperArm------------------------ -000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 -000.10 0000.00 0028.55 0079.09 -007.50 0001.00	upperArm2------------------------ 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0261.45 0079.09 0027.50 0001.00
foreArm-------------------------- -000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 -000.10 0000.00 0028.55 0134.09 0001.41 0001.00	foreArm2-------------------------- 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0261.45 0134.09 0018.59 0001.00

rollArm------------------------- -000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 -000.10 0000.00 0028.55 0166.06 0010.00 0001.00	rollArm2------------------------- 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0261.45 0166.06 0010.00 0001.00
pitchHand----------------------- -000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 -000.10 0000.00 0028.55 0185.66 0004.11 0001.00	pitchHand2----------------------- 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0261.45 0185.66 0015.89 0001.00
rollHand------------------------ -000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 -000.10 0000.00 0028.55 0190.66 0010.00 0001.00	rollHand2------------------------ 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0000.00 0000.00 0000.00 0000.10 0000.00 0261.45 0190.66 0010.00 0001.00

After the matrix is obtain, then this matrix can be implemented into the robot by using a programming language.

A. Designing the Movements of Each Robot Parts

The inverse kinematic robotics problem has been the focus of kinematic analysis for robot manipulators. In order to determine all possible formations to place the end effector of a robot manipulator at a particular point in space, we must compute the movements associated with each joint variable.

Actually, there is a wide range of movement by a robot to pick up and release an object. Movements is well designed in order to become more efficient robot motion. Here the researchers also designed a robot movement to pick up and release an object. Several variations of the robot motion is:

1. MovingPose.

MovingPose is the movement of the robot where the robot will be back at the starting position when it would pick up and release an object. To view the programming code can be seen in the picture below.

```
public bool MovingPose(
{
define the sequence for bNext 1 and 2 are false

define target 1 and 2  as the initial pose

calculate distance 1 with subtract the target 1 with
angle of foreArmMesh;
calculate distance 2 with subtract the target 2 with
angle of upperArmMesh;
if distance 1 more than and equal to 0.2 float
{
if target1 less than and equal to angle of foreArmMesh;
calculate angle of foreArmMesh at z point;
}
else bNext1 is true;
if distance2 more thatn and equal to 0.2float
{
if target2 less than and equal to angle of upperArmMesh;
calculate angle of upperArmMeshat z point;
}
else bNext2 is true;
return bNext1 and bNext2;
}
```

Figure 4.8 MovingPose Code

2. WasitJointRotate

WaistJointRotate is a movement where the waist will turn towards the goal of the robot to pick up an object. To view the programming code can be seen in the picture below.

```
public bool WaistJointRotate(TVMesh targetMesh)
{
Define target position;
Define waist joint calculation;
Define bResult is false
If {waist calculation more than and 0,1)
{ shoulder mesh rotate Y axis to the left as 0.21}
else
define bResult is true;
return bResult;
}
}
```

Figure 4.9 WaistJointRotate Code

3. ReachShoulder1st

ReachShoulder1st is the movement of the robot where the robot move the shoulder first to pick up and release an object. To view the programming code can be seen in the picture below.

```
public bool ReachShoulder1st(TVMesh targetMesh)
{
Define target (0f, 0f, 0f));

define bResult1 is false;
define bResult2 is false;

input dShoulderJointAngle as
ShoulderJointAngleCalculation(target);
if (dShoulderJointAngle more than 0.1f)
{
m_upperArmMesh Rotate Z axis to the left as 0.12f, false}
else bResult1 is true;

if (bResult1)
{
input dElbowJointAngle as
ElbowJointAngleCalculation(target);
if (dElbowJointAngle more than 0.1f)
{
m_foreArmMesh Rotate Z axis to the left as 0.12f false);
}
else bResult2 is true;
}
return (bResult1 & bResult2);
}
```

Figure 4.10 ReachShoulder1st Code

4. ReachElbow1st

ReachElbow1st is the movement of the robot where the robot move the elbow joint first to pick up and release an object. To view the programming code can be seen in the picture below.

```
public bool ReachElbow1st(TVMesh targetMesh)
{
Define target position(0f, 0f, 0f));

define bResult1 is false;
define bResult2 is false;

input dElbowJointAngle as
ElbowJointAngleCalculation(target);

if (dElbowJointAngle more than 0.1f)
{
 m_foreArmMesh Rotate Z axis to the left as 0.12f,
false);
}
else bResult1 is true;

if (bResult1)
{
float dShoulderJointAngle =
ShoulderJointAngleCalculation(target);

if (dShoulderJointAngle more than 0.1f)
{
m_foreArmMesh Rotate Z axis to the left as 0.12f,
false);

}
else bResult2 is true;

}

return (bResult1 & bResult2);
}
```

Figure 4.11 ReachShoulder1st Code

5. ReachSimultantToolVertical

ReachsimultantToolVertical is the movement of the robot where the shoulder, elbow and tool moving simultaneously in a vertical to pick up and release an object. To view the programming code can be seen in the picture below.

```
public bool ReachSimultanToolVertical(TVMesh targetMesh)
{
Define position(0f, 0f, 0f));

define bResult is false;

input dShoulderJointAngle as
ShoulderJointAngleCalculation(target);
input dElbowJointAngle as
ElbowJointAngleCalculation(target);

if ((dShoulderJointAngle more than 0.1f) &
(dElbowJointAngle more than 0.1f))
{
dShoulderJointAngle =
ShoulderJointAngleCalculation(target);
if (dShoulderJointAngle more than 0.1f)
m_foreArmMesh Rotate Z axis to the left as 0.12f,
false);

ApproachVerticalDown();

dElbowJointAngle as ElbowJointAngleCalculation(target);
if (dElbowJointAngle more than 0.1f)
 m_foreArmMesh Rotate Z axis to the left as 0.12f,
false);
ApproachVerticalDown();

}
else bResult is true;

return bResult;
}
```

Figure 4.12 ReachSimultantToolVertical

B. Flow Diagram

Below is a flow diagram of a robot in a pick up or release the object.

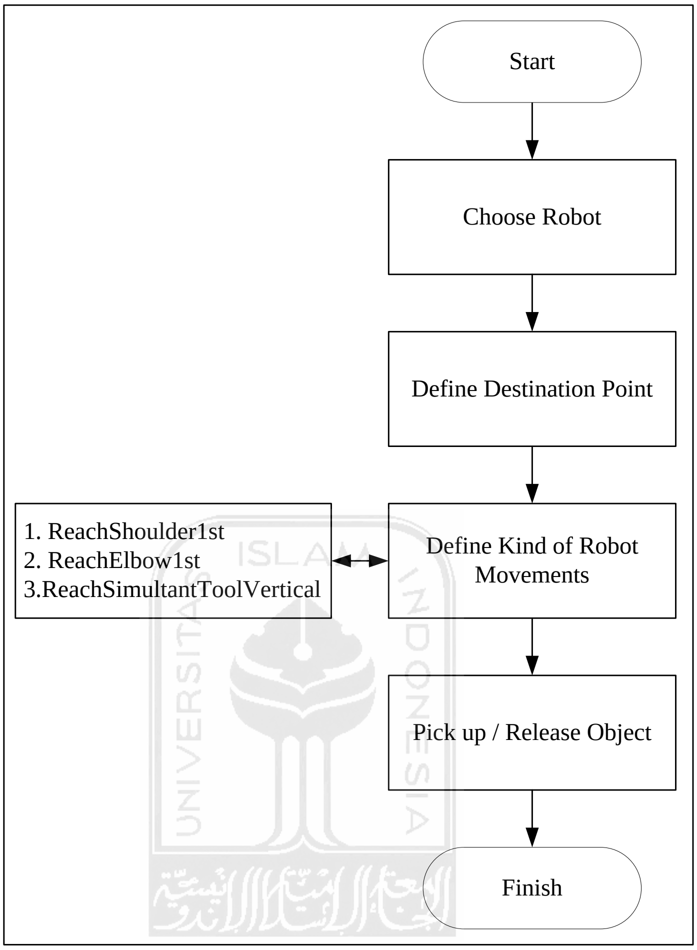


Figure 4.13 Flow Diagram of Robot Assignment

To assign the task to the robot, there are some things to do. The first is selecting the robot, in this study there are two robots that are used. The second is to determine the destination coordinates, this is done so that the end effector of the robot knows where to move. The third is to determine what to use movement, this movement aims to be more efficiently used depending on the location of the object to be retrieved. The fourth is to take or remove the object, the robot used in this study were assigned only to pick up and release objects as a function of material handling.

C. SequenceJob

SequenceJob is a whole sequence of scenarios that have been implemented into the program are made. Starting from Robot 1 took the first point on the raw material storage 1 using ReachShoulder1st movement type and move toward the second point on the conveyor 1 and release the raw material from the type of movement ReachShoulder1st, then the raw material will be moved by the conveyor 1 to the third point. After the raw material up to the third point, the robot 2 will take the raw material from the type of movement ReachSimultantToolVertical then moves toward the fourth point on Ncdrilling, on Ncdrilling raw material will be processed into finished product. After a finished product, robot 2 will take the finished product by using this type of movement ReachSimultantToolVertical then move toward the point of the fifth and finished product release on the conveyor 2. Then the finish product will be driven by the conveyor 2 to the sixth point, after finishing sixth product to the point, robot 1 will take the finished product by using this type of movement and put the finished product ReachElbow1st to seventh points on storage 2 using the type of movement ReachElbow1st.

CHAPTER V

DISCUSSION

5.1 Testing of Robot Model

To make sure the robot is actually able to pick up objects on the target that has been determined it is necessary to test, testing is done by manually measuring the position of the cylinder. After the measurements will be compared with that shown on the display screen. Y position of the base of the cylinder is 20, if added to the height of the cylinder is 10 then the y position of the cylinder is 30. Y position of cylinder on the display is 32, this is because it provides a space so that the end effector of the robot does not enter into the cylinder. To determine the performance of a model, the model will be tested. Testing the model in order to test whether the robot can move according to the procedure. Testing models of the robot is done manually, by pressing keys 0-9, O and P keys on the keyboard. The program code used to run a virtual robot arm, can be seen in Figure 5.1.

```
public void Check_JointRotation()
        {
        //waist joint
       (1)  if (input key(input key 1))
            {
               shoulderMesh.RotateY(1f, true);
            }
       (2)  if (input key(input key 2))
            {
               shoulderMesh.RotateY(-1f, true);
            }
```

```
//shoulder joint
    (3)  if (input key(input key 3))
         {
            upperArmMesh.RotateZ(1f, true);
         }
    (4)  if (input key(input key 4))
         {
            upperArmMesh.RotateZ(-1f, true);
         }
//elbow joint
    (5)  if (input key (input key 5))
         {
            foreArmMesh.RotateZ(1f, true);
         }
    (6)  if (input key(input key 6))
         {
            foreArmMesh.RotateZ(-1f, true):
  //roll arm
    (7)  if (input key (input key 7))
         {
            rollArmMesh.RotateY(1f, true);
         }
    (8)  if (input key(input key 8))
         {
            rollArmMesh.RotateY(-1f, true);
         }
  //pitch hand
    (9)  if (input key (input key 9))
         {
            pitchHandMesh.RotateZ(1f, true);
         }
    (10) if (input key (input key 0))
         {
            pitchHandMesh.RotateZ(-1f, true);
          }
   //roll hand
    (11) if (input key (input key P))
          {
             rollHandMesh.RotateY(1f, true);
          }
    (12) if (input key (input key O))
          {
             rollHandMesh.RotateY(-1f, true);
             }
          }
```

Figure 5.1 Testing of Robot Model Code

For an explanation of the first rule is if the 1 key on the keyboard is pressed then the waist joint will rotate in a clockwise direction. The second rule, if the 2 key on the keyboard is pressed then the waist joint will rotate counter clockwise. The third rule, if the 3 key on the keyboard is pressed then the shoulder joint will rotate in a clockwise direction. The fourth rule, if the 4 key on the keyboard is pressed then the shoulder joint will rotate counter clockwise. The fifth rule, if the 5 key on the keyboard is pressed then the elbow joint will rotate in a clockwise direction. The sixth rule, if the 6 key on the keyboard is pressed then the elbow joint will rotate counter clockwise. Seventh rule, if the button is pressed the keyboard 7 what will roll arm rotates clockwise. Eighth rule, if the button is pressed on the keyboard 8 then roll arm will rotate counter clockwise. The ninth rule, if the button 9 on the keyboard is pressed the pitch of the hand will rotate in a clockwise direction. Tenth rule, if the 0 key on the keyboard is pressed the pitch of the hand will rotate counter clockwise. Eleventh rule, if the P button is pressed on the keyboard then roll the hand will rotate in a clockwise direction. Twelfth rule, if the O button on the keyboard is pressed then roll the hand will rotate counter clockwise.

CHAPTER VI

CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions

The conclusions obtained from this study is:

Build and control system to control the VM models done by the inverse Kinematic method and then designing the movement of the model so the model can move according to what is desired. VM development system using TV3D as engine and C# as programming language, it will set to form the virtual environment which interact with the user.

6.2 Recomendations

Some suggestions for further research in order to perfect the system is:

1. NC Drilling in this study have not been working properly, for further research, it should be programmable so that the VM is more perfect.

2. The system should be constructed using an estimate of the time so it is useful to regulate the production scheduling.

3. System built would be better if future studies using the "interface" that aims to make the setting of production scheduling becomes easier.

4. Modular modelling and programming for flexible extension.

REFERENCES

Amos. H.C.Ng, Adolfsson. J, Sundberg. M, De Vin. L.J, (2008). Virtual manufacturing for press line monitoring and diagnostics. *International Journal of Machine Tools & Manufacture*, 48 no;5, 565–575.

Bruno. S & Luigi. V, (2000). An inverse kinematics algorithm for interaction control of a flexible arm with a compliant surface. *Control Engineering Practice*, 9; 191-198.

Chua. C.K, Leong. K.F, Lim. C.S, (2003). *Rapid prototyping: pronciples and applications*. World Scientific. Denver.

Fok. S. C, Lim. S. Y. E, Chan. G. Y. C, (2009). An intelligent virtual environment to support cognitive rehabilitation: relearning meal preparation skills for independent living. *Biomechatronics and Biomedical Robotics*, 1 no. 1.

Fu. K.S, Gonzales. R.C, Lee. C.S.G, (1988). Robotics, Control, Sensing, Vision, and Intelligence. McGraw-Hill. NewYork.

Fumarola. M, Poelman. R, (2011) .Generating virtual environments of real world facilities: Discussing four different approaches. *Automation in Construction*, 20 no.3; 263-269.

Groover. M.P, Weiss. M, Nagel. R.N, Odrey. N.G, (1986). *Industrial Robotics Technology, Programming, and Application*. McGraw-Hill, inc. New York.

Hongbo.L (2009). Web-based rapid prototyping and manufacturing system : A review. *Computers in Industry*, 60 no.9; 643-656.

Khan. A.W, Raouf. A, Cheng. K, (2011). *Virtual Manufacturing*. Springer. New York.

Koren. Yorem, (1987). *Robotics for Engineering*. McGraw-Hill, inc. New York.

Lee W.B, Cheung C.F, Li J.G (2001). Applications of virtual manufacturing in material processing. *Journal of material Processing Technology*, 113 no.1-3; 416-423.

Offodile. O.F, L. Layek, Malek. Abdel, (2002). The virtual manufacturing paradigm: The impact of IT/IS outsourcing on manufacturing strategy. *Int. J. Production Economics*, 75; 147–159.

Park Sang C, (2005). A methodology for creating virtual model for a flexible manufacturing system. Computers in Industry. *Computers in Industry*, 56 no.7; 734–746.

Prashant. B, & Zetu .D, (2001). *Virtual Manufacturing*. John wiley & sons, inc. New York.

Ronald. A. P, & James. C. B, (1996). A Novel Robust Algorithm for Controlling Robotic Manipulators. *Mechanical Engineering*, 333 no.1; 101-111.

Saadettin. K. S, Sanchoy. D. K, Zulal. Gungor (2010). A genetic algorithm based heuristic for scheduling of virtual manufacturing cells (VMCs). *Computers &Operations Research*, 37 no.6; 1148–1156.

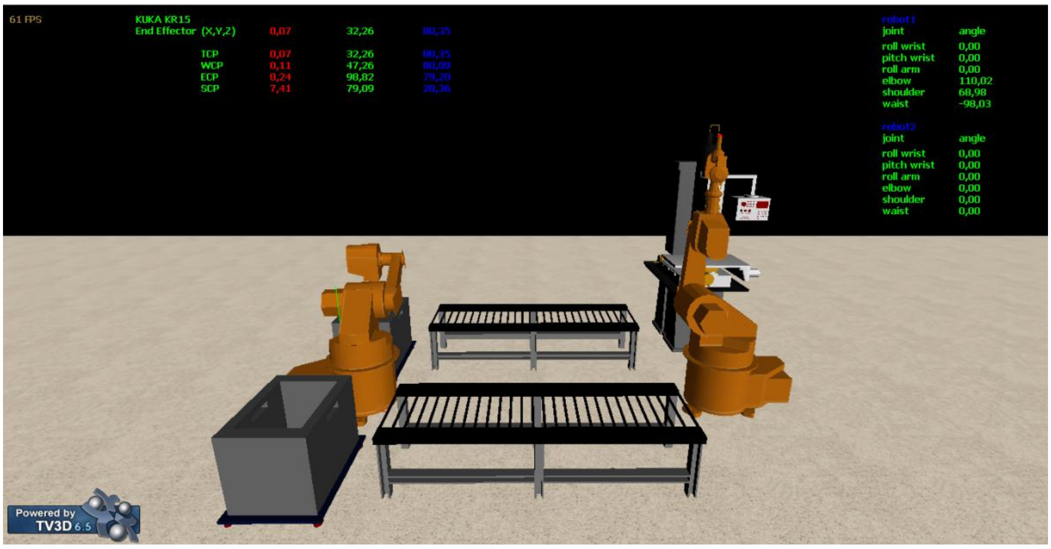
Slomp. J, Chowdary. B.V, Suresh. N.C (2005). Design of virtual manufacturing cells : a mathematical programming approach. *Robotics and Computer-Integrated Manufacturing*, 21 no.3; 273–288.

Sreenivas. T, & Subhash. K, (1999). A Novel Robust Algorithm for Controlling Robotic Manipulators. *Information Sciences*, 116; 147-164.

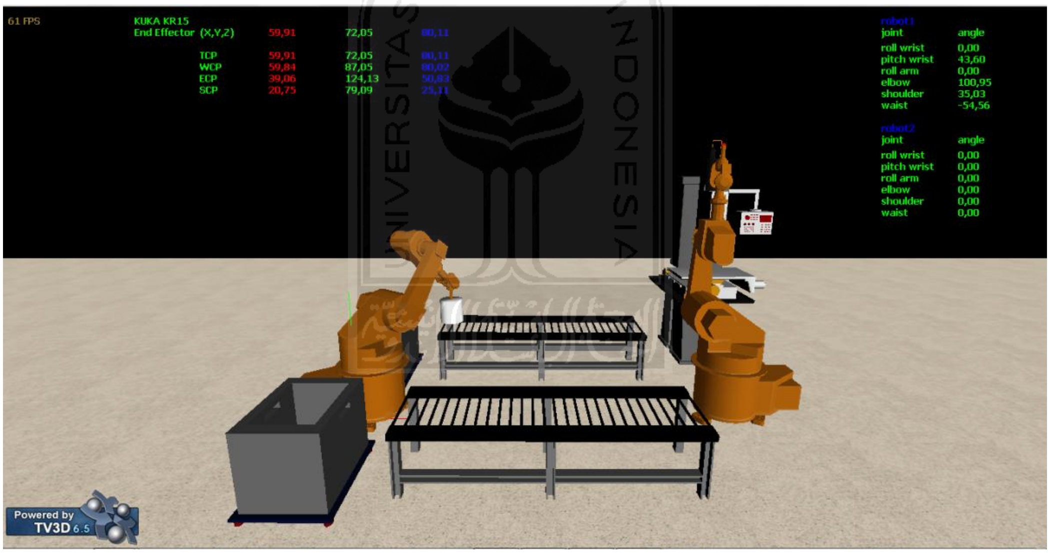
Xiangyu. W, & Jerry Jen. H.T, (2011). *Collaborative Design in Virtual Environments*. Springer.New York.

Xue. Yan, & P.Gue, (1996). A eview of rapid prototyping technologies and systems. *Computer-aided Design*, 28 no 4; 307-318.

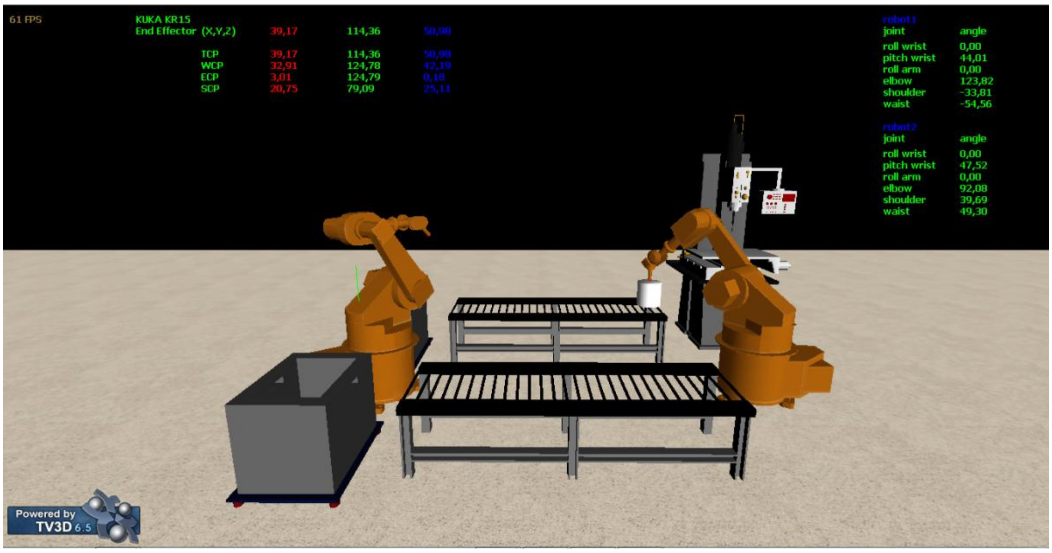
APPENDICES



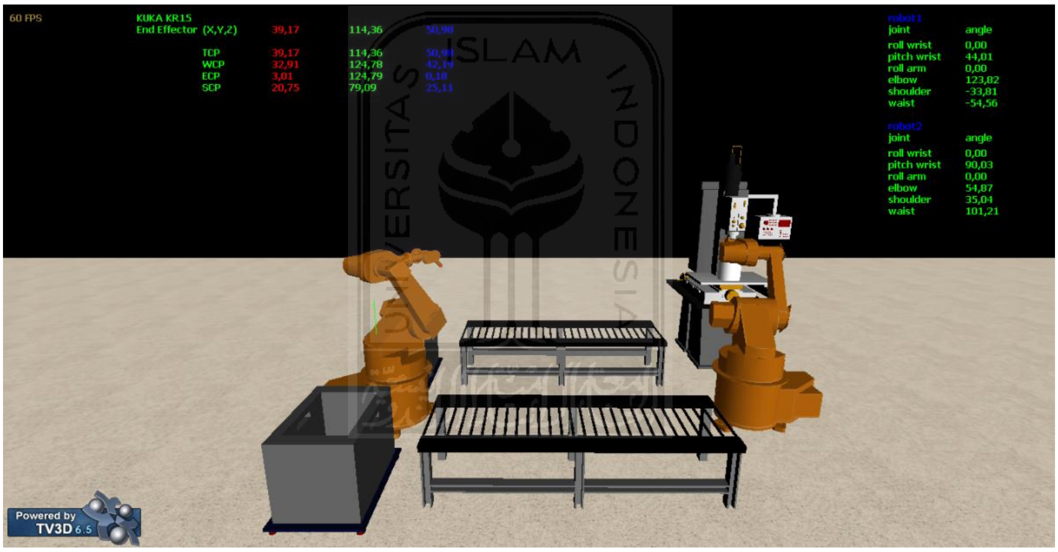
(a)



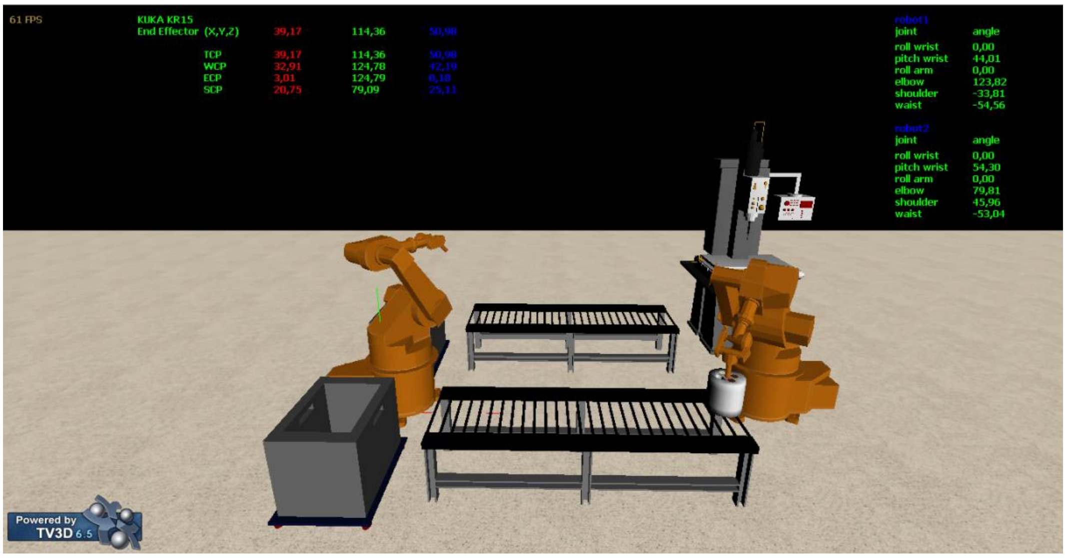
(b)


61 FPS
KUKA KR15
End Effector (X,Y,Z) 39,17 114,36 50,98
TCP 39,17 114,36 50,98
WCP 32,91 124,78 42,19
ECP 3,01 124,79 0,18
SCP 20,75 79,09 25,11
robot1
joint angle
roll wrist 0,00
pitch wrist 44,01
roll arm 0,00
elbow 123,82
shoulder -33,81
waist -54,56
robot2
joint angle
roll wrist 0,00
pitch wrist 47,52
roll arm 0,00
elbow 92,08
shoulder 39,69
waist 49,30
Powered by
TV3D 6.5

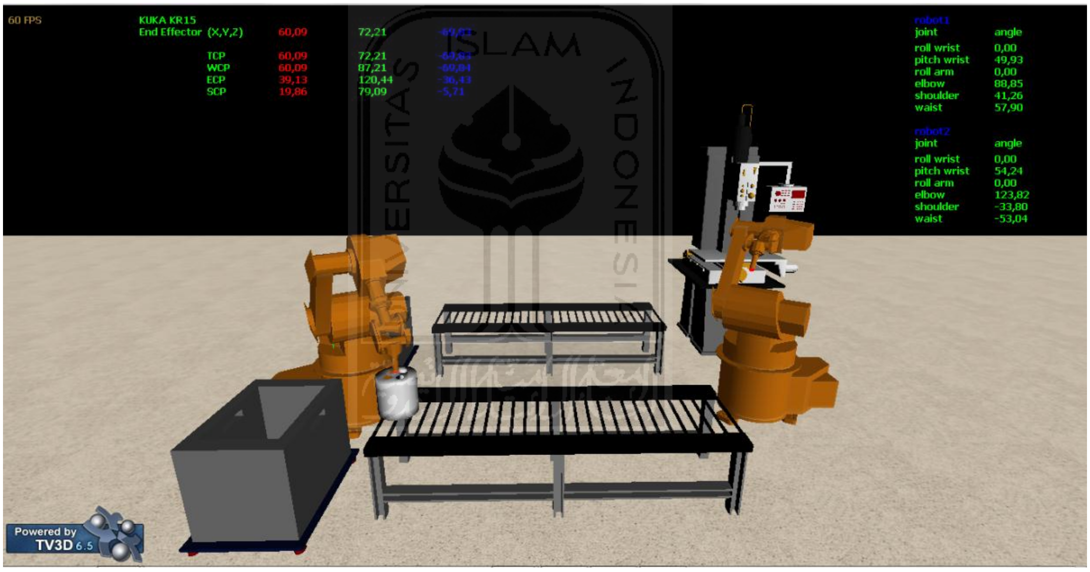
(c)


60 FPS
KUKA KR15
End Effector (X,Y,Z) 39,17 114,36 50,98
TCP 39,17 114,36 50,98
WCP 32,91 124,78 42,19
ECP 3,01 124,79 0,18
SCP 20,75 79,09 25,11
robot1
joint angle
roll wrist 0,00
pitch wrist 44,01
roll arm 0,00
elbow 123,82
shoulder -33,81
waist -54,56
robot2
joint angle
roll wrist 0,00
pitch wrist 90,03
roll arm 0,00
elbow 54,87
shoulder 35,04
waist 101,21
Powered by
TV3D 6.5
UNIVERSITAS ISLAM INDONESIA

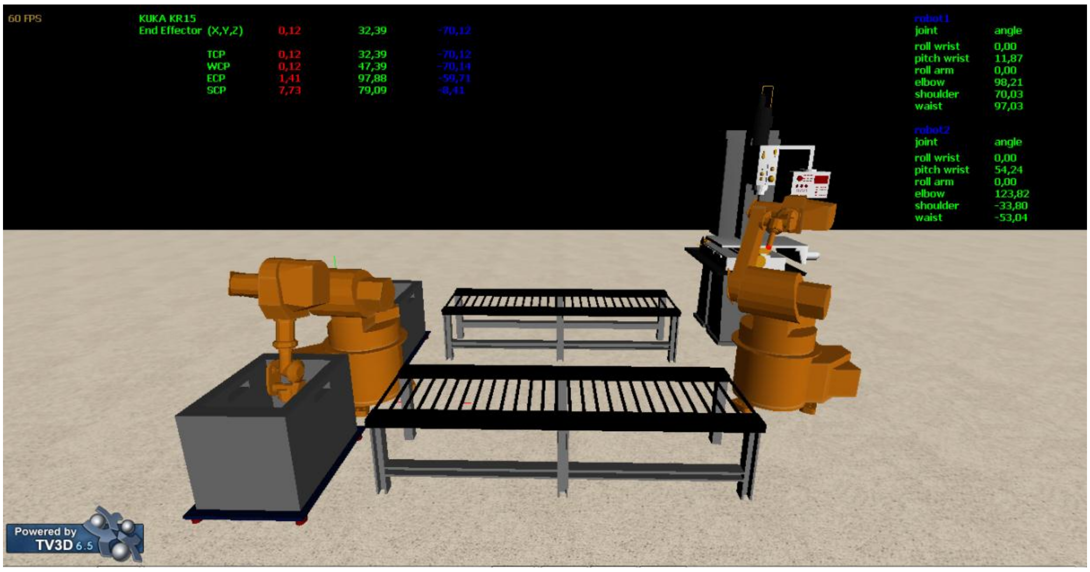
(d)


61 FPS
KUKA KR15
End Effector (X,Y,Z) 39,17 114,36 50,98
TCP 39,17 114,36 50,98
WCP 32,91 124,78 42,19
ECP 3,01 124,79 0,18
SCP 20,75 79,09 25,11
robot1
joint angle
roll wrist 0,00
pitch wrist 44,01
roll arm 0,00
elbow 123,82
shoulder -33,81
waist -54,56
robot2
joint angle
roll wrist 0,00
pitch wrist 54,30
roll arm 0,00
elbow 79,81
shoulder 45,96
waist -53,04
Powered by
TV3D 6.5

(e)


60 FPS
KUKA KR15
End Effector (X,Y,Z) 60,09 72,21 -69,83
TCP 60,09 72,21 -69,83
WCP 60,09 87,21 -69,84
ECP 39,13 120,44 -36,43
SCP 19,86 79,09 -5,71
robot1
joint angle
roll wrist 0,00
pitch wrist 49,93
roll arm 0,00
elbow 88,85
shoulder 41,26
waist 57,90
robot2
joint angle
roll wrist 0,00
pitch wrist 54,24
roll arm 0,00
elbow 123,82
shoulder -33,80
waist -53,04
Powered by
TV3D 6.5

(f)



(g)

a,b,c,d,e,f,g All Process of VM System

Programming Code

```
#region ------ DECLARE OBJECTS FIELD ------
        //mesh
        private TVMesh _roomMesh;
        private TVMesh _floorMesh;
        private TVMesh _cylinderMesh;
        private TVMesh[] _BoxMesh = new TVMesh[2];
        private TVMesh _NcMesh;
        private TVMesh _targetObjectMesh;
        private TVMesh _ObjectMesh;
        private TVMesh[] _conveyorMesh = new TVMesh[2]; //conveyor
        private TVMesh _pointMesh = new TVMesh();

        //home position
        private TV_3DVECTOR _HOMEpos;
        private TV_3DVECTOR _HOMEposDir;

        //end effector position, direction
        private TV_3DVECTOR _endEffCurrentPos;
        private TV_3DVECTOR _endEffCurrentDir;
        private float _endEff2WorldOrigin;

        private TV_3DVECTOR _targetPos;
        private TV_3DVECTOR _targetPosDir;

        //cylinder position
        private TV_3DVECTOR _ptA0 = new TV_3DVECTOR(60, 40, 80);
        private TV_3DVECTOR _ptA1 = new TV_3DVECTOR(60, 70, 80);

        private TV_3DVECTOR _ptB0 = new TV_3DVECTOR(220, 50, 80);
        private TV_3DVECTOR _ptB1 = new TV_3DVECTOR(220, 70, 80);

        private TV_3DVECTOR _ptC0 = new TV_3DVECTOR(0, 20, 70);
```

```csharp
        private TV_3DVECTOR _ptC1 = new TV_3DVECTOR(0, 30, 70);

        private TV_3DVECTOR _ptD0 = new TV_3DVECTOR(300, 100, 110);
        private TV_3DVECTOR _ptD1 = new TV_3DVECTOR(300, 107, 110);

        private TV_3DVECTOR _ptE0 = new TV_3DVECTOR(220, 20, -70);
        private TV_3DVECTOR _ptE1 = new TV_3DVECTOR(220, 70, -70);

        private TV_3DVECTOR _ptF0 = new TV_3DVECTOR(60, 50, -70);
        private TV_3DVECTOR _ptF1 = new TV_3DVECTOR(60, 70, -70);

        private TV_3DVECTOR _ptG0 = new TV_3DVECTOR(0, 20, -70);
        private TV_3DVECTOR _ptG1 = new TV_3DVECTOR(0, 30, -70);

        private TV_3DVECTOR _cylinderPosition = new TV_3DVECTOR();
        private TV_3DVECTOR _cylinderDestination = new TV_3DVECTOR();
        private TV_3DVECTOR _cylinderDirection = new TV_3DVECTOR();
        float _cylinderAngleY = new float();

        private TV_3DVECTOR _ObjectPosition = new TV_3DVECTOR();
        private TV_3DVECTOR _ObjectDestination = new TV_3DVECTOR();
        private TV_3DVECTOR _ObjectDirection = new TV_3DVECTOR();


        private TV_3DVECTOR dVector = new TV_3DVECTOR();
        private TV_3DVECTOR dV2 = new TV_3DVECTOR();

        //joint position, world coordinate
        private TV_3DVECTOR _worldOrigin = new TV_3DVECTOR(0, 0, 0);
        #endregion

    #region ------ DECLARE VARIABLES AND CONSTANT FIELD ------
        //directory
        private string _mediaDir = "D:\\RobotProject\\COMMON\\";

        //scale
        private float _scale = 0.1f;

        //simulation loop flag
        private bool _doLoop = true;

        //camera
        private float _cameraPosX;
        private float _cameraPosY;
        private float _cameraPosZ;
        private float _cameraLookAtX;
        private float _cameraLookAtY;
        private float _cameraLookAtZ;

        //table
        private TVMesh[] _ptMesh = new TVMesh[5];
        private float[] _ptAngleXZ = new float[5];

        private int _intRed = new TV_COLOR(1.0f, 0.0f, 0.0f, 1).GetIntColor();
        private int _intGreen = new TV_COLOR(0.0f, 1.0f, 0.0f, 1).GetIntColor();
        private int _intBlue = new TV_COLOR(0.0f, 0.0f, 1.0f, 1).GetIntColor();

        //cylinder dimension
        private float _radius = 10f;
        private float _height = 20f;

        private string _fmt = "0.#0;-0.#0";
```

```
        private float _fTime;            //tick value

        private bool _flagENTER = false;

        #endregion


#region ------- INITIATE FUNCTIONS FIELD -------

        private void InitEngine()
        {
            _engine.AllowMultithreading(true);
            _engine.SetDebugMode(true, true);
            _engine.SetDebugFile(Application.StartupPath + "\\debugfile.txt");
            _engine.Init3DWindowed(this.panel1.Handle);
            _engine.DisplayFPS(true);
            _engine.SetAngleSystem(CONST_TV_ANGLE.TV_ANGLE_DEGREE);
            _engine.SetVSync(true);

_engine.SetWatermarkParameters(CONST_TV_WATERMARKPLACE.TV_WATERMARK_BOTTOMLEFT,
0.001f);
        }

        private void InitScene(string strRenderMode)
        {
            if (strRenderMode.ToUpper() == "POINT")
                _scene.SetRenderMode(CONST_TV_RENDERMODE.TV_POINT);

            if (strRenderMode.ToUpper() == "LINE")
                _scene.SetRenderMode(CONST_TV_RENDERMODE.TV_LINE);

            if (strRenderMode.ToUpper() == "SOLID")
                _scene.SetRenderMode(CONST_TV_RENDERMODE.TV_SOLID);

            _scene.SetShadowParameters(new TV_COLOR(0f, 0f, 0f, 0.5f).GetIntColor(),
false);

        }

        private void InitInputs()
        {
            //input keyboard, mouse
            _input.Initialize(true, false);
        }

        //********************************
        public void InitMeshes()
        {
            robot1.SetName("robot1");
            robot1.SetPosition(10, 0, 10);
            robot1.SetRotateY(180f);
            robot1.SetScale(0.1f);
            robot1.InitializeRobot();

            robot2.SetName("robot2");
            robot2.SetPosition(280, 0, 10f);
            robot2.SetScale(0.1f);
            robot2.InitializeRobot();

            #region conveyor
            for (int i = 0; i <= 1; i++)
            {
```

```
                _conveyorMesh[i] = new TVMesh();
                _conveyorMesh[i] = _scene.CreateMeshBuilder("conveyor");
                _conveyorMesh[i].LoadXFile(_mediaDir + "kuka_kr15_x\\conveyor1.x");
                _texture.LoadTexture(String.Format(_mediaDir + "Textures\\black.bmp",
Application.StartupPath));

_conveyorMesh[i].SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
                _conveyorMesh[i].SetPosition(140.0f, 8.0f, -150.0f * (i + 1) +
200.0f);
                _conveyorMesh[i].SetScale(0.8f, 0.8f, 0.8f);
                _conveyorMesh[i].SetRotation(-90, 0, 0);

                #region box
                _BoxMesh[i] = _scene.CreateMeshBuilder("graybox");
                _BoxMesh[i].LoadXFile(_mediaDir + "kuka_kr15_x\\graybox.x");
                _BoxMesh[i].SetPosition(0, 0, 155.0f*(i+1)-200.0f);
                _BoxMesh[i].SetScale(0.8f, 1.1f, 0.8f);
                _BoxMesh[i].SetRotation(-90, 0, 0);

_BoxMesh[i].SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
                #endregion

            }
            #endregion


            #region cylinder
            _cylinderMesh = _scene.CreateMeshBuilder("Cylinder1");
            _cylinderMesh.CreateCylinder(_radius, _height, 30, true);//iPrecision!
jumlah sisi polygon dlm lingkaran, makin banyak makin halus
            _cylinderMesh.SetTexture(_globals.GetTex("RawTexture"));
            _cylinderMesh.SetMaterial(_globals.GetMat("basic Material"));
            _cylinderMesh.SetPosition(_ptC0.x, _ptC0.y + _height / 2, _ptC0.z);
            _cylinderMesh.SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            _cylinderAngleY =
_mathLibrary.Direction2Ang(_cylinderMesh.GetPosition().x,
_cylinderMesh.GetPosition().z);
            _cylinderMesh.Enable(true);
            #endregion

            #region NCDrilling
            _NcMesh = _scene.CreateMeshBuilder("ncDrilling");
            _NcMesh.LoadXFile(_mediaDir + "kuka_kr15_x\\ncdrilling.x");
            _NcMesh.SetPosition(300, 0, 140);
            _NcMesh.SetScale(0.11f, 0.11f, 0.11f);
            _NcMesh.SetRotation(-90, 180, 90);
            _NcMesh.SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            #endregion

            #region targetObject
            _targetObjectMesh = _scene.CreateMeshBuilder("targetObject");
            _targetObjectMesh.CreateSphere(1.5f);
            _targetObjectMesh.SetColor(_intRed);

_targetObjectMesh.SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            //_targetObjectMesh.SetRotation(180f, 0, 0); //Y axis down
            _targetObjectMesh.SetPosition(300f, 90f + 20f, 110f);
            _targetObjectMesh.SetScale(1f, 1f, 1f);
            _targetObjectMesh.Enable(false);
            #endregion

            #region Object
```

```
            _ObjectMesh = _scene.CreateMeshBuilder("Object");
            _ObjectMesh.LoadXFile(_mediaDir + "kuka_kr15_x\\whitecylinder.x");
           // _ObjectMesh.LoadXFile(_mediaDir + "kuka_kr15_x\\cylinderholes.x");
            _ObjectMesh.SetParent(CONST_TV_NODETYPE.TV_NODETYPE_MESH,
_targetObjectMesh.GetIndex(), 0);
            _ObjectMesh.SetPosition(0, 0, 0);
            _ObjectMesh.SetScale(1f, 1f, 1f);
            _ObjectMesh.SetRotation(90f, 0, 0);
            _ObjectMesh.SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            _ObjectMesh.Enable(false);
            #endregion



            #region point mesh
            _ptMesh[1] = _scene.CreateMeshBuilder("ptA");
            _ptMesh[1].CreateSphere(2.5f);
            _ptMesh[1].SetPosition(60f, 70f, 80f);
            _ptMesh[1].SetColor(_intRed);
            _ptMesh[1].SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            //_ptMesh[1].Enable(false);
            _ptAngleXZ[1] = _mathLibrary.Direction2Ang(_ptMesh[1].GetPosition().x,
_ptMesh[1].GetPosition().z);

            _ptMesh[2] = _scene.CreateMeshBuilder("ptB");
            _ptMesh[2].CreateSphere(2.5f);
            _ptMesh[2].SetPosition(220f, 70f, 80f);
            _ptMesh[2].SetColor(_intGreen);
            _ptMesh[2].SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            //_ptMesh[2].Enable(false);
            _ptAngleXZ[2] = _mathLibrary.Direction2Ang(_ptMesh[2].GetPosition().x,
_ptMesh[2].GetPosition().z);

            _ptMesh[3] = _scene.CreateMeshBuilder("ptC");
            _ptMesh[3].CreateSphere(2.5f);
            _ptMesh[3].SetPosition(220f, 70f, -70f);
            _ptMesh[3].SetColor(_intBlue);
            _ptMesh[3].SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            //_ptMesh[3].Enable(false);
            _ptAngleXZ[3] = _mathLibrary.Direction2Ang(_ptMesh[3].GetPosition().x,
_ptMesh[3].GetPosition().z);

            _ptMesh[4] = _scene.CreateMeshBuilder("ptD");
            _ptMesh[4].CreateSphere(2.5f);
            _ptMesh[4].SetPosition(60f, 70f, -70f);
            _ptMesh[4].SetColor(new TV_COLOR(0.2f, 0.5f, 0.1f, 1f).GetIntColor());
            _ptMesh[4].SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            //_ptMesh[4].Enable(false);
            _ptAngleXZ[4] = _mathLibrary.Direction2Ang(_ptMesh[4].GetPosition().x,
_ptMesh[4].GetPosition().z);


            _pointMesh = _scene.CreateMeshBuilder("ptE");
            _pointMesh.CreateSphere(2.5f);
            _pointMesh.SetPosition(0f, 20f, 70f);
            _pointMesh.SetColor(new TV_COLOR(0.2f, 0.5f, 0.1f, 1f).GetIntColor());
            _pointMesh.SetLightingMode(CONST_TV_LIGHTINGMODE.TV_LIGHTING_MANAGED);
            #endregion

            //write mesh matrix to a file
            if (_bWrite2File)
            {
```

```
                #region LINK MATRIX
                Write2File("**********************************", false);
                Write2File("LINK MATRIX", false);
                Write2File("**********************************", false);
                Write2File("world coordinate", true);
                Write2File("waist--------------------------",
robot1.m_waistMesh.GetMatrix());
                Write2File("shoulder-------------------------",
robot1.m_shoulderMesh.GetMatrix());
                Write2File("upperArm-------------------------",
robot1.m_upperArmMesh.GetMatrix());
                Write2File("foreArm--------------------------",
robot1.m_foreArmMesh.GetMatrix());
                Write2File("rollArm--------------------------",
robot1.m_rollArmMesh.GetMatrix());
                Write2File("pitchHand------------------------",
robot1.m_pitchHandMesh.GetMatrix());
                Write2File("rollHand-------------------------",
robot1.m_rollHandMesh.GetMatrix());
                Write2File("**********************************", false);
                Write2File(" ", false);

                #endregion
            }

        }
        //*******************************

        private void InitLights()
        {
            _lights.CreateDirectionalLight(new TV_3DVECTOR(-1f, -1f, 1f), 1f, 1f, 1f,
"frontLamp");
        }

        private void InitCamera()
        {
            _cameraPosX = 100;
            _cameraPosY = 100;
            _cameraPosZ = -250;

            _cameraLookAtX = 50;
            _cameraLookAtY = 50;
            _cameraLookAtZ = 250;

            _camera = _scene.GetCamera();
            _camera.SetViewFrustum(60, 1000, 0.1f);
            _camera.SetPosition(_cameraPosX, _cameraPosY, _cameraPosZ);
            _camera.SetLookAt(_cameraLookAtX, _cameraLookAtY, _cameraLookAtZ);
        }

        private void InitViewport()
        {
            _viewport = _engine.CreateViewport(this.Handle, "viewport1");
            _viewport.SetCamera(_camera);
        }

        private void InitTextures()
        {
            _texture.LoadTexture(_mediaDir + "Textures\\3.bmp", "RoomTexture", -1, -1,
CONST_TV_COLORKEY.TV_COLORKEY_NO, true);
            _texture.LoadTexture(_mediaDir + "Textures\\warna.bmp", "ArmTexture", -1,
-1, CONST_TV_COLORKEY.TV_COLORKEY_NO, true);
```

```
        }

        private void InitMaterials()
        {
            int idMat;
            idMat = _materials.CreateLightMaterial(1, 1, 1, 1, 0.015f, 1, "basic
material");
            _materials.SetEmissive(idMat, 0.24f, 0.24f, 0.24f, 1f);
            _materials.SetSpecular(idMat, 1f, 1f, 1f, 1f);
        }

        private void InitRoom()
        {
            _roomMesh = (TVMesh)_scene.CreateMeshBuilder("roommesh");
            _floorMesh = _scene.CreateMeshBuilder("floormesh");


            _floorMesh.AddFloor(_globals.GetTex("RoomTexture"), -5000.0f, -5000 - 0f,
5000.0f, 5000.0f, 0.0f, 100.0f, 100.0f, true);

        }
        #endregion
```