DYNAMIC COMPUTER SIMULATION OF PARTS FEEDING ON A VIBRATORY BOWL FEEDER

Patrick S. K. Chua and F. L. Tan
School of Mechanical and Production Engineering,
Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798
E-mail: mfltan@ntu.edu.sg

ABSTRACT

This paper describes the development of three-dimensional (3D) computer simulation software using Java 3D API to simulate parts feeding and orienting in a vibratory bowl feeder. The simulation software can perform dynamic simulation of part motion and the interaction of simple parts with the designed orienting mechanism, therefore enabling the reliability and capability of the design of the mechanism to be assessed prior to fabrication of the bowl feeder.

Keywords : *Java 3D, Parts Feeding, Simulation, Vibratory Bowl Feeder*

1. INTRODUCTION

In modern automated assembly lines, feeding of engineering parts in the correct orientation is one of the important processes and vibratory bowl feeding is the most common method used in the automated feeding and orienting of industrial parts. Vibratory bowl feeders are low cost and capable of comparatively diverse applications. Besides being extremely fast in their ability to orient complex parts, they are also reliable in operation.

As shown in Figure 1, a typical vibratory bowl feeder [1] consists of a bowl mounted on a base by three inclined leaf springs. In such a feeder system, the parts travel up the bowl along a helical track. The orienting mechanism near to the outlet of the track causes the undesirable parts to be re-oriented or rejected by falling off of the track. The result is that parts reaching the outlet of the bowl are correctly oriented. Besides being extremely fast in their ability to orient complex parts, they are also simple in construction and dependable in operation [2].

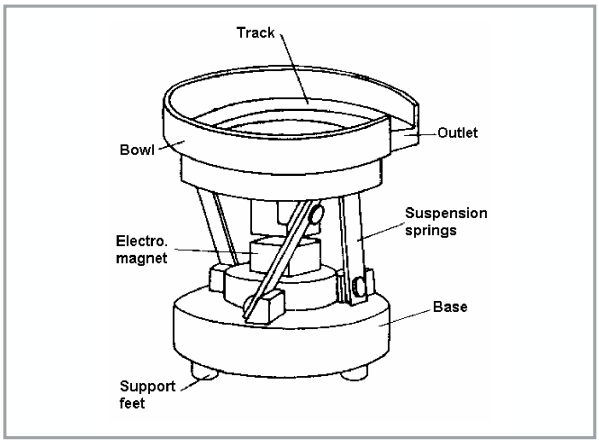

Figure 1: Typical vibratory bowl feeder

Unfortunately, vibratory bowl feeders are not flexible in that the same bowl cannot be used to feed different batches of parts differing in shapes or dimensions. For each new part, the orienting mechanisms have to be designed and they are developed by intuition and trial-and-error. This approach is unproductive, inefficient, expensive and time-consuming.

Designers of the orienting mechanism of the vibratory bowl feeder are searching for simulation softwares that can help them assess the performance of their design of the orienting mechanism before constructing a physical mechanism. If they are not satisfied with the performance, they can redesign the mechanism and assess it again using the simulation software until they are satisfied with its performance. Without the simulation software, they would have to physically construct the orienting mechanism. The construction of a physical orienting mechanism takes much time, effort, and skill, and if found to perform unsatisfactorily during testing would have to be disposed. The whole procedure of reconstruction and testing has to be repeated. This is a tedious, expensive and time consuming process and it also delays delivery time to customers.

There is, therefore, a need for the development of a simulation software to help bowl feeder designers to speed up the design of the orienting mechanism of the bowl feeder and to deliver reliable, efficient feeders to customers at short lead time and competitive price. The objective of this paper is to present a simulation software developed for the vibratory bowl feeder. The development of 3D computer simulation software is proposed for simulating parts feeding and orienting in a vibratory bowl feeder. The 3D simulation is a potentially powerful tool for the design and performance analysis of the orienting mechanism of the vibratory bowl feeder prior to fabrication. The computer simulation approach removes the disadvantages encountered in the conventional design of the vibratory bowl feeder. This study is based on the part motion and orientation on the track of vibratory bowl feeder.

Boothroyd et al. [3] presented a hierarchical structure coding system utilizing Group Technology. The coding system is designed to fulfill the functions of design aid and parts classification. The efficiency of feeding and orienting devices is entirely determined by the natural resting behaviour of a given part on the surface of a hopper. In 1972, Boothroyd et al. [4] proposed an energy barrier method that predicted the probabilities of the natural resting aspects of square prisms, rectangular prisms and cylinders with the analysis of the energy involved in these prisms changing their resting aspect from one to

another. In 1993, Ngoi et al. [5] proposed a new approach, called centroid solid angle (CSA) method, to predict the probabilities of the natural resting aspects of parts on soft surfaces.

As for simulation in parts feeding, researchers have proposed simulation as a technology for making the designer's job more efficient and effective. Jakiela and Krishnasamy [6] discussed a scheme for two-dimensional simulation of vibratory parts feeding. Caine [7] used the configuration space paradigm to develop an interactive system for simulating and manipulating designs. In 1997, Maul and Thomas [8] presented a system model of vibratory bowl feeder. Perhaps the closest to this study is the work of Berkowitz and Canny [9] who provided a comparison of real and simulated orienting design for cylindrical parts. For this project, computer simulation has been used for evaluating vibratory bowl orienting mechanism designs.

There are two major classes of simulation software: simulation languages and simulators [10]. The strength of simulation languages lies in their ability to model any kind of system, regardless of its complexity. Java 3D API [11] and Java platform were developed by Sun Microsystems to be able to provide the visualization, user interaction, and thread management for the computer simulation.

The software developed using Java technology provides solutions in a web-centric technology that delivers the key capabilities needed to find, manage and use information in a dispersed environment. As the growing trend for customization in product development such as vibratory bowl feeder increases, manufacturers need real-time design specifications, cost structure, and production capabilities across the entire product life cycle. For these reasons, the simulation software can play an important role in providing design solution.

2. ORIENTING MECHANISM FOR RECTANGULAR PART

In general, an orienting system for a vibratory bowl feeder consists of one or more orienting mechanisms arranged in series along the bowl track. These orienting mechanisms ensure that only the parts in the correct orientation pass to the chute. Parts in undesired orientation are either re-oriented to the desired orientation or are rejected.

As an example in this paper, the rectangular part will be studied. The dimensions of the rectangular parts studied with variation in length from 12 to 31 mm, variation in width from 8 to 12 mm and variation in height from 3 to 6 mm. The developed simulation software cannot handle circular parts or other parts with complex geometry. This will be left for future work. There are six stable block orientations for rectangular parts in a vibratory bowl feeder as shown in Figure 2: Flat Lengthwise (FL), On-Edge Lengthwise (OEL), Erect Lengthwise (EL), Flat Crosswise (FC), On-Edge Crosswise (OEC), and Erect Crosswise (EC). The desired behaviour of an orienting system is to uniquely orient rectangular parts beginning in one of above six orientations.

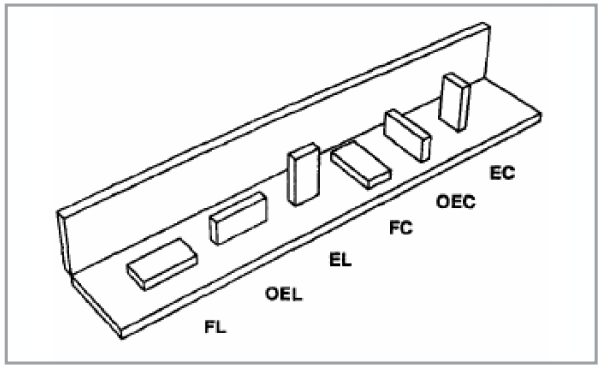

Figure 2: Rectangular part orientations

A typical series is shown in Figure 3 for the feeding and orienting of rectangular parts. From the knowledge of natural resting aspects, the probability of the parts in FL and FC orientations is the highest. An orienting mechanism design that allows only parts in orientation FL to pass to the outlet of the bowl will result in the highest feeding efficiency. Parts in orientation FC can be rejected by designing a narrow track as shown in Figure 3. The dimensions of the track can be changed in the simulation software to suit the parts going along the track. The width "b" of the narrow track can be of any value depending on the dimensions of the part studied. It is physically adjustable. The software will only allow parts in orientation FL (Figure 2) to pass through to the exit of the vibratory bowl. Thus, the FL orientation should be the desired orientation for high productivity and efficiency reasons. Parts in the other four orientations (i.e. OEL, EL, OEC and EC) can be rejected by the wiper blade.

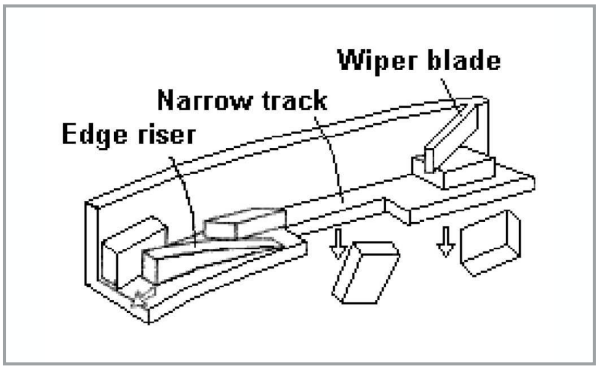

Figure 3: Orienting mechanisms

Data for building the simulation software were obtained by performing many hundreds of repeat tests on rectangular parts each with different lengths to achieve different b/w ratios and using different amplitudes of vibration of the vibratory bowl. For each rectangular part, the test was repeated 385 times for each amplitude in order to achieve a confidence level of 95% [12]. The test results are as shown in Figure 4. The simulation software uses the data obtained to set the condition of part motion whenever the feeding part encounters the narrow track.

3. COMPUTER SIMULATION SOFTWARE DEVELOPMENT

Currently the two most widely available programming for 3D simulation are VRML and Java 3D. While VRML is simple to use, it lacks any real programming capability. In this project, Java programming is chosen using Java 3D API.

3.1 BUILDING A SCENE GRAPH

A Java 3D virtual universe is created from a scene graph. A scene graph is created using instances of Java 3D classes. The scene graph is assembled from objects to define the geometry,

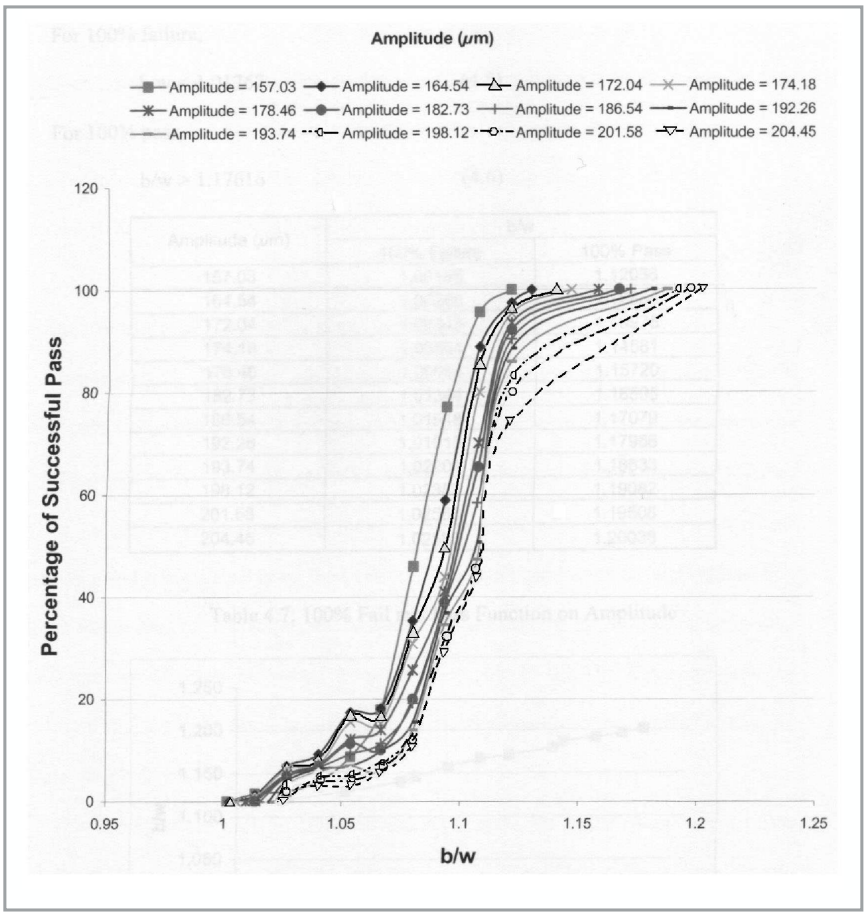

Figure 4: Pass rate tunction on b/w ratio at various amplitudes of vibration

sound, lights, location, orientation, and appearance of visual objects. A Java 3D scene graphs is constructed of Node objects in parent-child relationships forming a tree structure.

3.2 MODEL REPRESENTATIONS

In solid and geometric modeling, there are two major representation schemata: B-rep (boundary representation) and CSG (constructive solid geometry). B-rep represents a solid object by describing its surface, while CSG represents a solid object by a set-theoretic boolean expression of primitive objects. In Java 3D, the geometrical object is described by its surfaces.

3.3 COLLISION DETECTION TECHNIQUE

The problem of collision detection or contact determination between two or more objects is fundamental to computer simulation, physical based modelling and computer-simulated environments. In this project, an essential component of part motion planning and collision detection is a geometric reasoning system which can detect contacts between the part and the orienting mechanism in the track. Although it does not provide a complete solution to the path-planning problem, it serves as a good indicator to show that the part encounters the orienting mechanism, track or wall when an actual collision occurs. The collision response is application dependent. The collision detection is based upon using bounding volumes and spatial decomposition techniques.

3.4 CREATING A COLLISION DETECTION SYSTEM WITH JAVA3D

The 3D scenes require some degree of interaction between objects within the scene and between the scene and the user. Such interactions frequently need to be collision based. Java 3D provides an abstract Behaviour Node class which can catch events dispatched by the behaviour scheduler, where the specification of the events to be caught are referred to as WakeUps. In addition to specifying the triggering events, the Java 3D collision WakeUps contain methods to determine the collided-with object, the colliding object, the route from the top of the scene graph to the collided-with object (a SceneGraphPath) and the route to the colliding object. These allow the user to access any of the methods of the objects involved. Thus, there is a large amount of information available to the scene creator.

Collision detection needs to determine the position and information of any intersection between part and track. The track is vibrating due to the vibration of the bowl feeder and thus the part is moved forward on the track by bouncing on the track. In the simulation software, the motion parameters that need to be entered into the software are the initial location of the part and dimensions of the part (Figure 7) and the dimensions of the narrow track (Figure 8). The software will calculate the x, y, and z component of the part velocities (Figure 6). While the Java 3D collision detection system provides the scene programmer with a range of available information, the information is not enough for application in this project. Once collision has occurred, one normally needs to respond in some way. In this case, a usually important aspect of this response is to know the status of moving part such as location, velocity and acceleration.

The entire Java3D scene being viewed is embedded in a virtual world. However, each viewable object is normally held within a TransformGroup node and the coordinates of the object are relative to that node rather than to the virtual world. Fortunately, there is the ability to obtain the transformation matrix which provides the conversion for the TransformGroup from local coordinates to virtual world coordinates. This matrix can then be used to transform each point in the shape geometry to provide the virtual world coordinates of the shape. This assumes that the capability for retrieving this conversion matrix has been set for the TransformGroup.

While one collision event is being processed, no other collision based events can be detected. This means that a shape cannot move while in contact with some surface and colliding

with a raised edge (for example the track and wall). The contact with the surface produces a collisionMovement event which blocks the collision detection. In a similar way, in the animated scene, if a feeding part collides with the track, close to the wall, then if the scene gets updated again before the part has left the collision region, no second collision can be detected and so the part may pass through the second side (the wall). In order to prevent this, resetting the wakeup condition is essential.

3.5 PART MOTION IMPLEMENTATION

Collisions between parts and track or orienting mechanism are detected in order to perform part motion. In dynamics simulations, the simulation software is responsible for the resulting behaviour of the colliding objects, and will need to calculate new velocities, accelerations, etc. for the objects. Two approaches [13] to produce the desired realistic behaviour in dynamics simulation software are analytical methods, and non-analytical methods (penalty methods).

Collision response using penalty method involves inserting temporary springs between objects at the points of contact. The results are only approximations, and it is computationally expensive. Collision response using analytical methods involves formulating, and solving Newtonian dynamical equations (involving linear and angular velocities, momentum, accelerations etc.). The solutions give exact answers, and the equations require fewer time steps to solve. In this project, no deformation of objects is assumed, and thus the analytical methods are chosen to simulate the part feeding and orienting behaviour.

In Java 3D, Behaviour is a class for specifying animations of or interaction with objects. The behaviour can change virtually any attribute of an object. Once a behaviour is specified for an object, the Java 3D system updates the position, orientation, colour, or other attributes, of the object automatically. To specify a behaviour for an object, the programmer creates the objects that specify the behaviour, adds the object to the scene graph, and makes the appropriate references among scene graph objects and the behaviour objects.

4. SOFTWARE APPLICATION

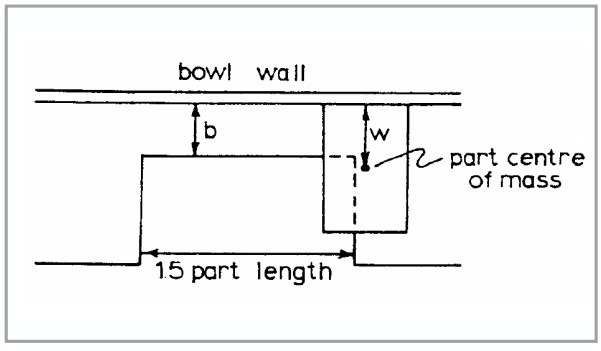

Figure 5: Design of narrow track

Based on system analysis and modelling, simulation software for parts feeding was developed. The software provides customised design setting, motion simulating and performance verification. As an example, the rectangle part shown in Figure 5 is simulated as moving on the narrow track with the customised settings in the simulation software.

The simulation software is intended for use on Java Virtual Machine (JVM) which is an abstract computing machine running on Windows98/NT/2000, Unix or Linux. A Pentium 166 MHz or faster processor with at least 32 megabytes of physical RAM is required to run the graphically based application. Sixty-four Megabytes of RAM is recommended. There should be 180 megabytes of free disk space before attempting to run Java3D using Sun's Java development and runtime software, such as Java 2 SDK Standard Edition, Java 2 Runtime Environment, Java3D API, Java 3D Runtime Environment, and Java Help. All above-mentioned software can be downloaded from the website [14].

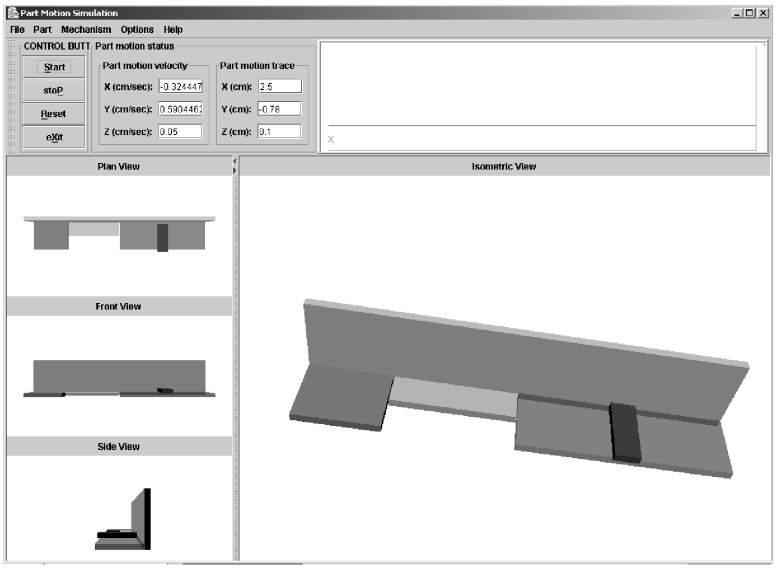

Figure 6: Main window

At the launching of the software, the main window will appear as shown in Figure 6. The main window contains three areas: the command area, the simulation area and performance report area.

By default, the command area appears at the top of the screen when the software is started. All commands are contained in the menu bar and the toolbar.

- The menu bar displays all the commands available in the software.
- The control button panel in the toolbar displays buttons that are shortcuts for commonly used menu commands.
- All commands support hot key operation.

The toolbar includes the performance report area as well. The part motion status panel reports the velocity and location of the feeding part, while the display area shows the part feeding trace visually. The third part is the simulation area below the toolbar. The right portion in Figure 7 is the isometric view of part simulation and there are three views which represent observation from top, front and side.

In order to start simulation for customised part and mechanism, the properties of part and mechanism need to be set in the software. It can be done by reading resource file or manually set in parameter setting panel. The part attributes will be set in the Part Definition window. Figure 7 shows rectangular part setting frame. The part size is designed at Rectangle Dimension panel and the Part Location panel is used to set initial position of feeding part. The orientation of the rectangular part is selected in Part Orientation panel.

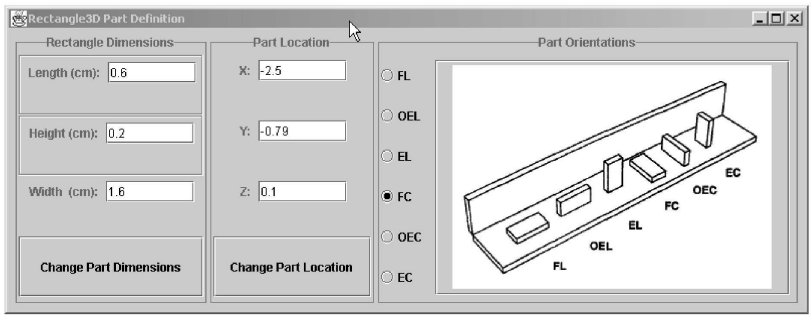

Figure 7: Part definition window

Similarly, the Mechanism Definition window is utilised to set the narrow track as shown in Figure 8. The scene graph will be refreshed using new values as soon as the "Change …" button is pressed.

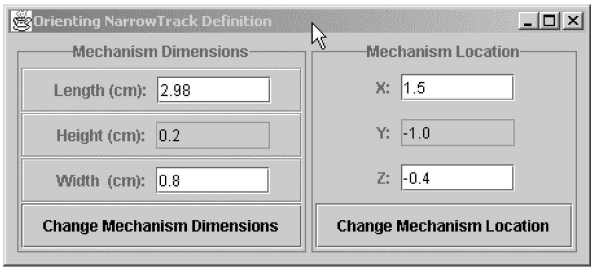

Figure 8: Narrow track definition window

Figures 9(a) and 9(b) show that a part with FC orientation will be rejected by the narrow track while Figure 9(c) and 9(d) show that the part with FL orientation can pass through the narrow track.

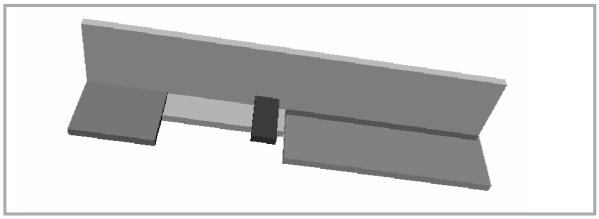

Figure 9(a): Part entering the narrow track

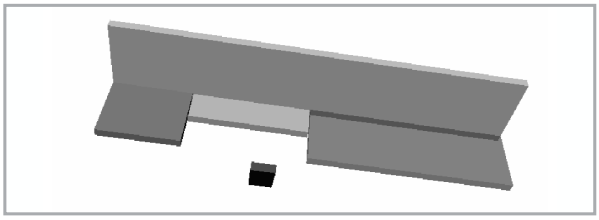

Figure 9(b): Part dropping off the narrow track

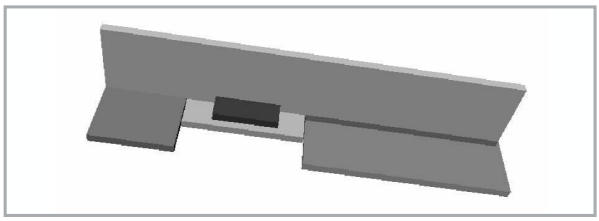

Figure 9(c): Part moving along the narrow track

5. CONCLUSION

The paper describes the development of Java 3D computer simulation software for simulating the motion of simple feeding part and its interaction with the orienting mechanism designed. The software enables the reliability and capability of the design of the orienting mechanism to be assessed prior to fabrication of the mechanism. The designer can then use the design information, which is saved in the resource file by the software, to generate the engineering drawings for fabrication. ■

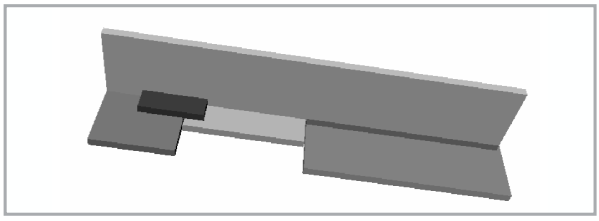

Figure 9(d): Part exiting from the narrow track

Figure 9: Simulation of rectangular part moving on the narrow track

REFERENCES

[1] G. Boothroyd, C.R. Poli, and L.E. Murch, Automatic Assembly (Manufacturing engineering and materials processing), New York: M. Dekker, 1982.

[2] F. Riley, Assembly Automation: A Management Handbook, Industry Press Inc., pp. 115-163, 1983.

[3] G. Boothroyd, C.R. Poli, and L.E. Murch, Handbook of Feeding and Orienting Techniques for Small Parts, Department of Mechanical Engineering, Uni-versity of Massachusetts, Amherst, MA, 1978.

[4] G. Boothroyd, A.H. Redford, C.R. Poli, and L.E. Murch., "Statistical Distributed of Natural Resting Aspects of Parts for Automatic Handling", Manufacturing Engineering Transactions, Vol. 1, pp. 99-105, 1972.

[5] B.K.A. Ngoi, S.S.G. Lee, and L.E.N. Lim, "Analyzing the Probabilities of the Natural Resting Aspects of a Component with a Displaced Center of Gravity", International Journal of Production Research, Vol. 33, No. 9, pp. 2387-2394, 1995.

[6] M. Jakiela, and J. Krishnasamy, "Computer simulation of vibratory parts feeding and assembly", Proceedings of 2nd International Conference on Discrete Element Methods, 1993.

[7] M.E. Caine, "The Design of Shape from Motion Constraints", Massachusetts Institute of Technology, Cambridge, MA. Ph.D. Thesis, Department of Mechanical Engineering, 1993.

[8] G. Maul, and M. Thomas, "A Systems Model and Simulation of the Vibratory Bowl Feeder", Journal of Manufacturing Systems, Vol. 16, No. 5, pp. 309-314, 1997.

[9] D.R. Berkowitz, and J. Canny, "A comparison of real and simulated designs for vibratory parts feeding", Robotics and Automation. Proceedings, IEEE International Conference, Vol. 3, pp. 2377–2382, 1997.

[10] A.M. Law, and W.D. Kelton, Simulation Modeling and Analysis, Second Edition, New York: McGraw-Hill, pp. 236-237, 1991.

[11] H.A. Sowizral, The Java 3D API specification, The Java Series Mass.: Addison-Wesley, 1998.

[12] J. Khazanie, Statistics: In a World of Applications, New York: Harper Collins College Publishers, , USA, 1996.

[13] D. Baraff, "Analytical methods for dynamic simulation of non-penetrating rigid bodies", Computer Graphics, Vol. 23, No. 3, pp. 223-231, 1989.

[14] Java Website, http://java.sun.com/products/. (Dec 2005)

PROFILES

Patrick Chua Soon Keong
School of Mechanical and Production Engineering,
Nanyang Technological University
50 Nanyang Avenue, Singapore 639798

Tan Fock Lai
School of Mechanical and Production Engineering,
Nanyang Technological University
50 Nanyang Avenue, Singapore 639798