〔非公開〕

ＴＲ－Ｃ－００８３

Interactions between objects in a virtual space

アラン　シャネゾン　　竹村　治雄　　北村　喜文
Alain CHANEZON　Haruo TAKEMURA　Yoshifumi KITAMURA

１９９３．１．２５

ＡＴＲ通信システム研究所

Interactions between objects in a virtual space

Alain CHANEZON, Haruo TAKEMURA,
and Yoshifumi KITAMURA

January 1993

ABSTRACT

This report describes a method to enable a manipulator in a virtual world to release an object on a surface without any force feed-back tool. The system does not require any command to learn and works in real time.

When a grabbed object comes too close to another, it is attracted. A distance between objects was designed to enable this calculation. It was based on a distance between faces. But the number of faces of the world being to big to reach real time calculation, simplifications had to be done. We designed "attracted faces" as faces on which an object can lie and "attracting faces" as faces which can attract objects. The calculation using only these faces can be done in real time.

Thanks to this interface, a manipulator who grabs an object can move it on some predefined planes (as for some example the top of a table) and control the position where he releases it. A first experiment showed that the interface is all the more useful than the action has to be precise.

Contents

1 Introduction

The department Artificial Intelligence of ATR is working on the design of a new teleconferencing system with realistic sensations using a virtual world. A 3D-picture is displayed for each person depending on his(her) view-point and a shared environment, where everybody can grab and move objects, is created.

To reach the goal (realistic sensations), it is important that people do not wear any tool of communication with the computer. Consequently the use of dataglove, 3D-glasses or force feedback tools is impossible.

For some actions, force feedback can be replaced by improving the human-computer interface. The goal of our research is to enable a manipulator to release an object on a surface.

We imagined two kinds of interface. In the first one, a command ("put *this item* on *that item*... please") is proposed to the user. This command can be an oral order or a special hand gesture. In the second one, the computer decides which is the intention of the manipulator.

One drawback of the first method is that the command has to be learned before a manipulator can use it. Keeping in mind the teleconferencing system, we decided that the new interface should be accessible to any user. Consequently we did not keep this method and chosed the second one (Figure 1).

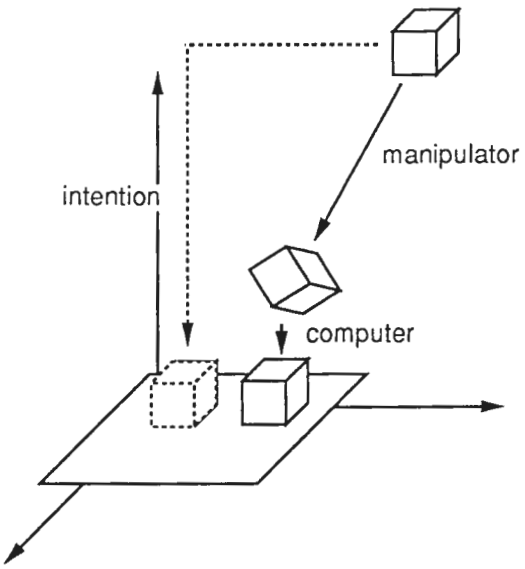

Figure 1: The interface

2 A first implementation

In our experiment, to move objects in the virtual world, the manipulator uses a dataglove and a Polhemus sensor. The dataglove gives informations about the shape of the hand. The shapes that can be detected are "grab" when the hand is closed and "release" when the hand is open. The Polhemus sensor gives the position (3 coordinates) and the orientation (the 3 Euler angles: azimuth,elevation and roll) of the hand in the space.

When an object has been grabbed, moved and released the computer decides thanks to its new position if the intention of the manipulator was to release it on an other object. If such is the case, it calculates a new position on the selected object.

2.1 The choice of an object

2.1.1 Need for a distance between faces

To select an object, the following assumption is made:

> a manipulator willing to put an object on a planar surface will put it as close as possible to the surface.

We had consequently to design a *distance between objects* to select the closest one. In this case, "close" means that the two external surfaces of the objects are close. A distance between barycenters could not have fitted our goal.

Considering that an object in a virtual space is designed as a collection of faces describing its external shape, the distance between two objects is the minimal distance between every couple of faces of these objects. A *distance between faces* had to be designed.

2.1.2 The distance implemented

A simple distance between barycenters could lead to big mistakes because close faces can have far-away barycenters. We tried to design a distance which should be as close as possible to the real distance between two faces.

The general idea is to consider that, the surfaces being planar, one of the two point s where the distance is minimum must be on an edge of its surface, the other being anywhere on its surface.

One face is considered as a collection of verteces and edges but no inside and a first distance is calculated. Then the same calculation is done with exchanged roles. The minimum distance is kept.

The function doing these calculations is *dist_f1_proj(f1,f2,..)* where *f1* designs the face of verteces and edges.

- First, the verteces of *f1* (M_i) are projected on the plane of *f2*. The distances of projection ($d_1(M_i)$) are calculated. If *f1* is intersecting the plane of *f2*, let $[M_iM_{i+1}]$ and $[M_jM_{j+1}]$ be the two edges intersected:

The two verteces of intersection I_1 and I_2 are added to the list of projected verteces,

$[M_i M_{i+1}]$ is separated into two edges $[M_i I_1]$ and $[I_1 M_{i+1}]$,

$[M_j M_{j+1}]$ is separated into two edges $[M_j I_2]$ and $[I_2 M_{j+1}]$.

- Then, for each projected vertex, the distance to *f2* is calculated (d_2). Three configurations are available (Figure 2).

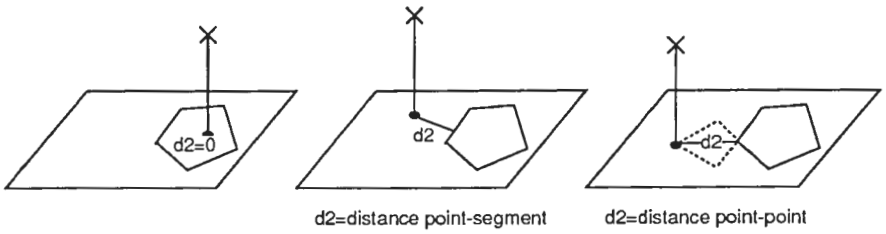

Figure 2: distance vertex-face

- Finally, for each projected edge, the intersection with the edges of *f2* and the verteces which should be closer to *f2*'s edges than the extremities are computed (Figure 3).

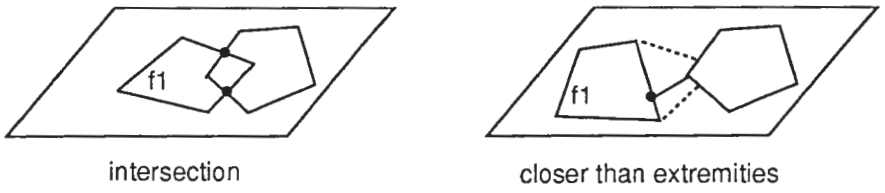

Figure 3: calculations on edges

d_2 is easily calculated for these verteces. d_1 on one vertex has to be found back thanks to the distances d_1 at the extremities of the segment.

The distance is the minimum of $\sqrt{d_1^2 + d_2^2}$ for all verteces which have been calculated.

2.1.3 A problem of coordinates

The organisation of the objects in the virtual world is a tree structure, where the link "father-child" means that if the father is moved, the child has to be moved the same way. The list of children of a father are given thanks to the link "brother" (Figure 4).

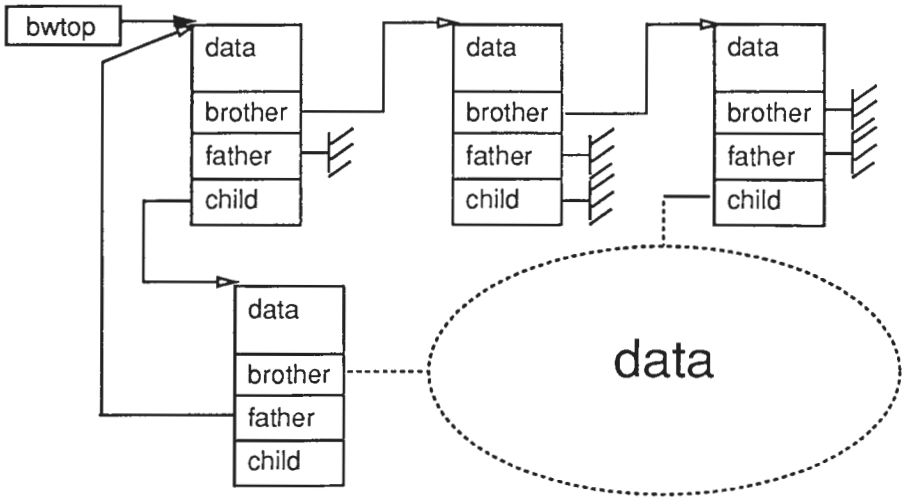

Figure 4: The structure of the world

An object is described as a collection of verteces which coordinates are independant from its position and orientation. These two caracteristics are added to the data describing the object. To enable the link "father-child" they are given in the base of the father. The result is that it is impossible to calculate the distance between two verteces of two different objects without finding a common ancester.

To solve such a problem of coordinates, we decided to add a matrix to the data describing an object (*bw_base*, which is the matrix of transformation of coordinates from the base of the object to the base of the toplevel father.

$$B \stackrel{\text{def}}{=} \underbrace{\underbrace{R_x R_y R_z T}_{object}\ \underbrace{R_x R_y R_z T}_{father} \ldots \underbrace{R_x R_y R_z T}_{bwtop}}_{bw\_base} \tag{1}$$

Applying this matrix to the coordinates of a vertex enables to have its new coordinates in a base common to all objects. As the scale is not included, it has to be added when the new coordinates are calculated.

2.1.4 Threshold or collision

to determine if the grabbed object has to be moved upon the selected object, two ways can be designed:

- a *threshold* value is used. If the distance is under this threshold then the grabbed object is moved. As a first step, we gave an arbitrary value

to this threshold because our goal was to see how quick the system was responding.

- with the distance implemented, the collision between two objects is easily detected: the distance is equal to zero. The choice to move or not can be done if there is a *collision*.

2.2 The movement

To move the grabbed object toward the selected one, we need to chose a couple of faces which should be attached. The movement of the grabbed object is then calculated to fulfil this goal.

To chose these faces, the following assumption is made:

> A manipulator grabbing an object and willing to release it on a planar surface will present to this surface the face he intends the object to rest on.

Consequently the selected couple is the couple of closest faces. It has already been calculated in the choice of the two closest objects.

2.2.1 Rotation

The first transformation made on the grabbed object is a rotation to have the two chosen faces parallel. The rotation is made around an axis defined by

- a vertex: the center of gravity of the selected face of the grabbed object
- a normalised vector which has the same direction as the common axis to the two planes of the two selected faces. It is the outer product of the normal vectors to the two faces.

The angle of rotation is calculated thanks to the dot products of the normal vectors to the two faces. The following constatation enables to find the right sign for the rotation:

> If the two normal vectors are pointing the same direction, the rotation around their outer product has to be positive to have the two faces parallel (Figure 5).

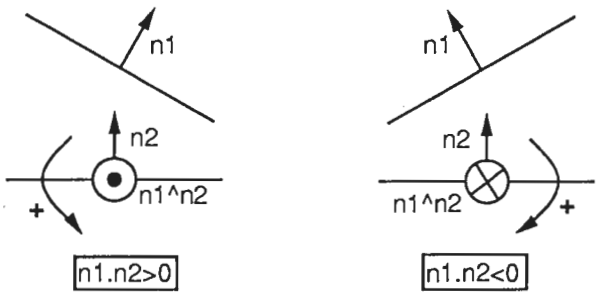

Figure 5: Rotation to have two faces parallel

The calculation of the matrix of rotation is made this way:

Let $P \stackrel{\text{def}}{=} (x, y, z)$ and $\vec{v} \stackrel{\text{def}}{=} (v_0, v_1, v_2)$ be the vertex and vector defining the axis of rotation and θ the angle of rotation, the matrix of rotation R is given by the following formula [1]:

$$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -x & -y & -z & 1 \end{pmatrix} A \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ x & y & z & 1 \end{pmatrix} \tag{2}$$

$$A = \begin{pmatrix} v_0^2 + (1 - v_0^2)cos\theta & v_0 v_1(1 - cos\theta) + v_2 sin\theta & v_0 v_2(1 - cos\theta) - v_1 sin\theta & 0 \\ v_0 v_1(1 - cos\theta) - v_2 sin\theta & v_1^2 + (1 - v_1^2)cos\theta & v_1 v_2(1 - cos\theta) + v_0 sin\theta & 0 \\ v_0 v_2(1 - cos\theta) + v_1 sin\theta & v_1 v_2(1 - cos\theta) - v_0 sin\theta & v_2^2 + (1 - v_2^2)cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

2.2.2 Translation

The second transformation made on the object is a translation which goal is to have the two objects attached.

The translation used is orthogonal to the face of the chosen object.

But another simple translation could be used, the one which changes the barycenter of the grabbed object in the barycenter of the chosen object. This would be interesting for some applications like playing chess. Indeed the manipulator wants to release his piece in the center of a face and not anywhere on the board.

2.2.3 Change basis

To display the virtual world, the program is using the Graphic Library of the Silicon Graphics computer [4]. To go from the object coordinate system to the

eye coordinate system a "ModelView" matrix has to be calculated. It is the product of a "Modelling" matrix (M) and a "Viewing" matrix (V).

$$\underset{\substack{eye\\coord}}{\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix}} = \underset{\substack{object\\coord}}{\begin{pmatrix} x \\ y \\ z \end{pmatrix}} \quad MV$$

The viewing matrix is defined thanks to the position of the eyes, given by a Polhemus sensor. It does not depend on the objects and their positions in the world but on the position of the observer.

The modelling matrix enables to manipulate an entire object keeping its own coordinate system. It is a combination of translations and rotations around the axes of the basis. In our program M is defined for an object as follows:

$$M = \underbrace{S}_{scale} \underbrace{R_x R_y R_z T}_{object} \underbrace{R_x R_y R_z T}_{father} ... \underbrace{R_x R_y R_z T}_{bwtop} \tag{3}$$

where S is a scale matrix (defining the expension of the object along the three axis). according to our definition of bw_base (equation 1 in section 2.1.3)

$$M = SB$$

When the object is moved by the computer, a new modelling matrix M' has to be computed. As the rotation and the translation are calculated in the base common to all objects, the new modelling matrix has the following form:

$$M' = SB(RT)$$

R being (See equation 2) the rotation and T the translation calculated by the computer. Let X be the whole matrix of transformation:

$$X \stackrel{\text{def}}{=} RT \tag{4}$$

To express the new position of the object as three rotations and a translation and consequently to fit the structure already designed for describing a position, we had to calculate another matrix X' such as:

$$M' = SBX = SX'B$$

$$\Rightarrow X' = BXB^{-1}$$

Then integrate X' in the matrix B as follows:

$$X'B = \underbrace{X' \underbrace{R_x R_y R_z T}_{object}}_{R'_x R'_y R'_z T'} \underbrace{R_x R_y R_z T}_{father} ... \underbrace{R_x R_y R_z T}_{bwtop} = B'$$

(See Appendix I for the calculation of the inverse of B and Appendix II for the transformation of $X' R_x R_y R_z T$ into three rotations and one translation.)

2.3 The result

The calculation of the closest object and the movement (if it was decided) was done when the manipulator released the grabbed object in this first version.

With a simple world composed of 27 cubes, the time between the releasing and the movement could be detected by the eye (approximatly 1 second).

With a more complicated world containing complex shapes made of hundreds of faces, we could go and have 2 or 3 coffees before the movement was calculated.

This is due to the fact that the number of distances to calculate grows with the number of faces of the world and the grabbed object. If the world has N faces and the grabbed object P faces, this number is $2P(N-P)$ (the 2 is because for each couple of faces the minimum between two distances is calculated). As an example, a space shuttle with 400 faces released in a world containing 2400 faces requires 1,600,000 calculations of distances!

Such a tool is completely unuseful if it can not be used in real time because it deteriorates the interface human-computer.

As we could not count on an improvement of the speed of the computer to solve our problem, considering that the number of faces defining a world will grow the same time, we had to find a way to lower the number of calculations of distances in the choice of an object.

3 A second implementation

One of the possibility to lower the number of calculations would be to make a more simple model for each object, as for exemple a cube, or a 6 faces block. But, as we mentionned before, even for a simple world with 27 cubes, the tool did not fit.

3.1 Simplifications

We did three kinds of simplifications. The first is to consider only close objects in the calculation (using a rough test), the second is to use only some faces and the third to simplify the distance between two faces.

3.1.1 Remove obviously far objects

If we can say with a very simple test that an object is far from the grabbed object, then there is no hope it will be chosen. To do so, we define for each object the "ray", which is the biggest distance between the center of gravity and any vertex of the surface.

Before calculating the precise distance between any object and the grabbed object, the distance between their barycenter is computed. If it is bigger than the sum of the rays plus the threshold value then the object is not considered in the calculation (Figure 6).

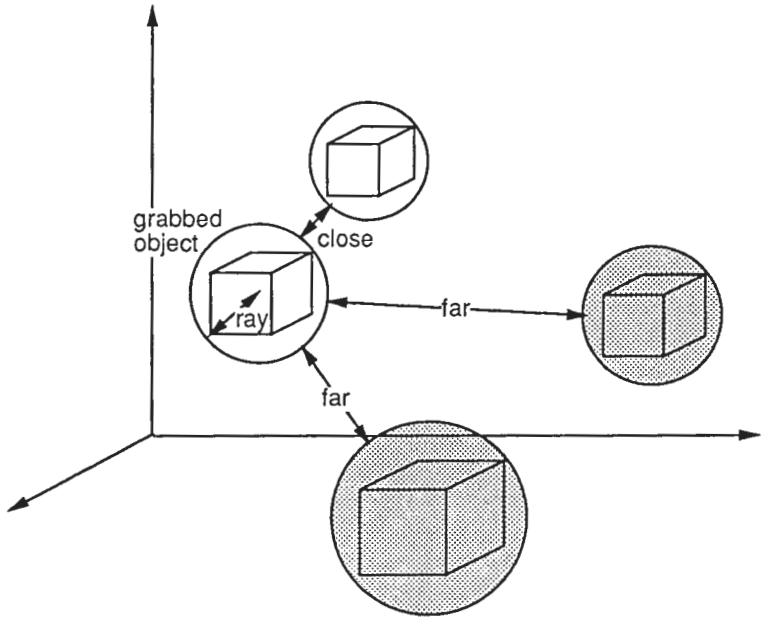

Figure 6: Remove far objets

3.1.2 Keep only important faces

Due to its shape, an object has some faces more important than others regarding the interaction with the world.

- *the grabbed object*: when somebody wants to release a cup on a table, a small number of positions are available. The cup has to be stable. It is not the case in the virtual world but a cup released with an unstable position would look strange to the eye of the manipulator. We gave the example of the cup, but it is true for all objects that a small number of faces are implied in the action of being released on a plane.

  Consequently, we defined *"attracted faces"* for each objects, as faces which can be attracted by the rest of the world when the object is grabbed. The other faces of the object do not enter anymore in the choice of an object to move toward.

- *the rest of the world*: Let us consider one more time the exemple of the cup and the table. Releasing his cup, the person will surely do it on the top of the table. The corners and the legs are surely not used to release anything on. In the virtual world, the top face only can be kept to be eventually chosen as a face to move toward.

  Considering that the same kind of consideration can be done for all objects in the world, we defined *"attracting faces"* as faces which can attract a grabbed object.

The "attracted faces" could be calculated automatically with a software designed to sort stable faces. Such a software is being implemented in ATR [2]. However, "attracting faces" can not be designed automatically without considering too many faces. These faces depend on the application developped thanks to the virtual world. For exemple, in a teleconferencing system, the ceiling and the walls of the room do not need to be attracting. There is no object which could be interesting to put on. But in the case of designing one's kitchen thanks to the virtual world, these same surfaces have to be attracting, to enable to fix a cupboard on a wall or a lamp on the ceiling. It is the work of the designer of the world to define these faces.

When an object is grabbed, only "attracted faces" on this object and "attracting faces" on the objects of the rest of the world are considered in the calculation.

3.1.3 Direction of attraction

Another simplification we made was to define in which direction an "attracted face" can be attracted and an "attracting face" can attract. This direction is given by the normal vector to the surface. This one is pointing to the inside or to the outside of the object (to the barycenter or in the opposite direction).

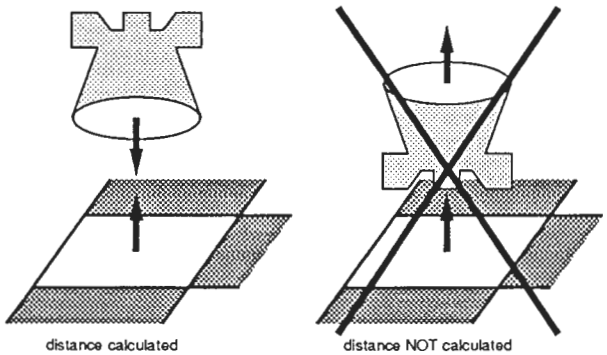

Figure 7: Direction of attraction

The distance between two faces is only calculated if their two normal vector are pointing opposite direction, which means that the angle made by these two vector is bigger than 90 degrees (Figure 7).

3.1.4 A new distance

The first distance implemented was unnecessarily complicated. The collision detection (see section 2.1.4) was impossible with the new version because a small number of faces only was used for calculation. We decided to simplify the calculation of the distance between two faces.

In the new distance, edges are not considered. First, one face is considered as a collection of verteces. The distance from these verteces to the other face is calculated by projecting them and then calculating the distance in the plane. The minimum for all verteces is kept. Then the roles of the two faces are exchanged.

When projecting the verteces of the "attracted face", if there is no projected verteces inside the face, the movement is not done (it means that after the movement, the two objects will not be attouched).

3.2 A real-time tool

3.2.1 Result of simplifications

With the modifications which have been made, the result was quick enough to implement it in real time. This means that instead of waiting the manipulator to release the object to make the calculations and eventually move the object, it is made at every instant when an object is grabbed (at the same rate as the refreshing of the display, this is to say nearly 10 times per second).

3.2.2 A problem of position

In the program of ATR, when the cube is grabbed and moved, the displaying of the cube is using the position of the object and the hand at the instant of grabbing and the new position of the hand. But the position and rotation arguments of the object are not changed until it is released (Figure 8)

The problem is that the program which calculates the interaction needs the object to be at its real position. Consequently, when the object is moved, its position and orientation must move the same way and not only when it is released.

This is what we did, but another problem occured. The displaying needed to have the original position and orientation of the cube. We added some variables to keep these values in memory during the movement.

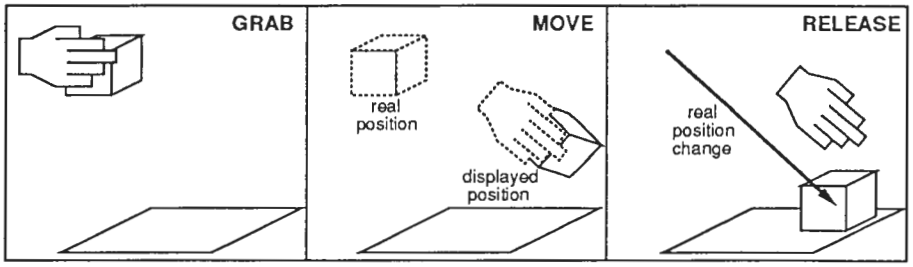

Figure 8: Position of an object when it is moved

3.2.3 The final tool

With the new tool, a manipulator can grab an object, move it to an "attracting plane", then if he keeps an "attracted face" of the object in a certain area around the "attracting plane", he can move the object on the plane (which is to say control 2 translations and 1 rotation -Figure 9).

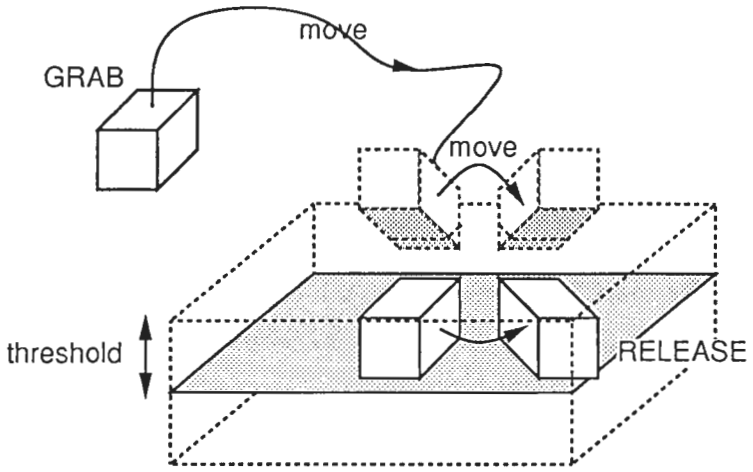

Figure 9: The interface

4 Two experiments

We did two experiments with the new interface. The first one was designed to find a good value for the threshold. The second one had for goal to evaluate the

amelioration of the performances of the manipulator thanks to the interaction between objects.

4.1 Find a threshold

4.1.1 Experiment method

Five persons took part to this experiment. Among them, some were very familiar with moving objects in the virtual space, some had no prior experience of those kind of manipulation. Consequently, everybody was asked to grab, move and release some objects until they felt comfortable before the experiment began.

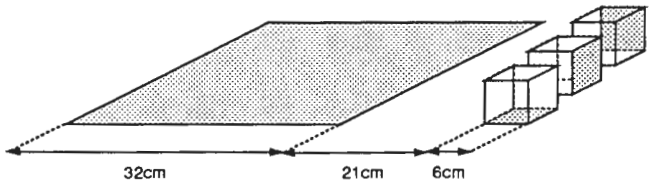

Figure 10: Experiment 1

For decreasing threshold values (from 32cm to 0.2cm), the subject had to move five times three predefined "attracted faces" of three different cubes on an attracting plane. The plane was 32cm large and the cubes had a 6cm edge. The cubes were 21cm distant from the plane (Figure 10). The task was considered to be achieved for a cube if the distace between its "attracted face" and the plane was less than 0.2cm. The manipulator was informed through a visual feedback (the cube changed color).

4.1.2 Results

For each person, two graphs could be obtained (Figure 11)

- The first graph is the mean time (made on 5 trials) to succeed to release one, two and the three cubes on the plane.

  The graph has nearly the same shape for everybody (a curve decreasing quickly at the beginning, then reaching a converging line) with some accidents sometimes. The two main accidents are the following:

  First, for some people the first value is a bit lower than the second which means that they took more care with the threshold 0.2 than with 0.25. The reason is that the threshold 0.2 was done last. A phenomene of fatigue and boredom is introduced. As these values are very close, it is not so important to know which one has been done the quickest.

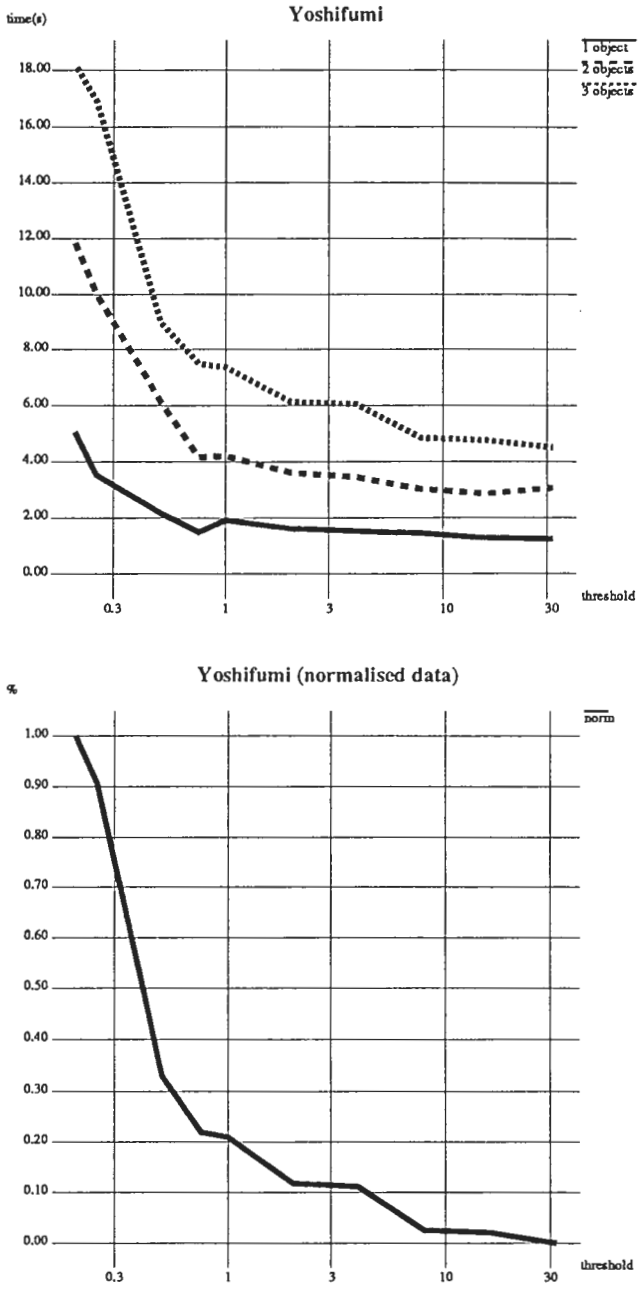

Figure 11: experimental result for one experimentator

Second, for some people the value for 32 is bigger than the one for 16. The reason is that the threshold 32 is done first. People starting the experiment do not dare move very quickly. Then, after 5 or 6 cubes, they start accelerating, eventhough the threshold decreases.

The following table gives for the five experimentators the variation of the time to succeed the action for the different values of the threshold.

experimentator	lowest	highest
no1	1.95	12.60
no2	3.43	30.85
no3	4.76	18.66
no4	4.63	34.19
no5	3.16	31.70

The scale is very different from one experimentator to the other. This is due to the fact that people had not the same experience to move objects in a virtual world.

Consequently we had to normalise data before the mean could be taken. To do so, we considered that the lowest time was nearly equal to the converging value and that the highest time was the time needed to accomplish the task without interaction and reduced the interval to the interval [0, 1] with the following formula:

$$x_{(\%)} = \frac{t - t_{min}}{t_{max} - t_{min}}$$

- The second graph is the time to move the three cubes on the plane normalised as described earlier (the result is a percentage: 100% correspond to no gain in comparison with the world without interaction and 0% to the maximum gain possible).

For the five experimentators, the results are gathered in the graphs given by figure 12.

The first one gives the normalised data of the five experimentators, and the second one is the mean curve of the five previous one. This one is used to find a good threshold value.

The interaction is introducing a discontinuity in the movement of the object (when the computer changes the position).This discontinuity looks strange to the manipulator, so it is important to lower its effect. To do so, the threshold value has to be as small as possible.

We chose a threshold which could enable the manipulator to gain 90% of what he could gain thanks to interaction. This threshold is at the intersection of the mean curve and the line y=0.1. As the experiment we did was not accurate enough (only five persons took part), we could only find an approximation of the best value and we chose 2 for the threshold.

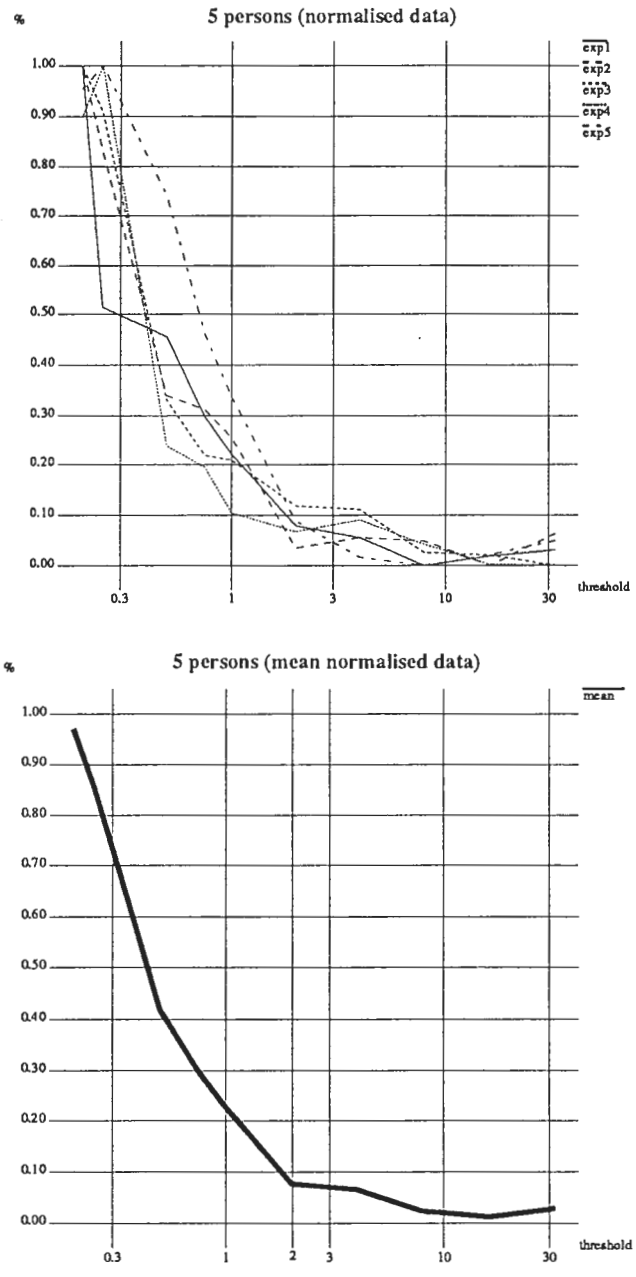

Figure 12: Eperimental results for the 5 manipulators

4.2 Evaluate the new tool

4.2.1 Experiment method

Five persons were asked to do ten times the same experiment. The goal was to move six times (three levels of difficulty times two configurations -with and without interaction-) a cube containing an "attracted face" on a fixed cube containing an "attracting face". The cubes were distant of 19cm and had a 6cm edge (Figure 13 left). To complete the task, the manipulator had to put the two interacting faces in contact and make the corners of the cubes coincide. To test the coincidence, the sum of the distances between all four corners of the "attracted face" and the closest corner of the "attracting face" was computed (Figure 13 right). A threshold enabled to change the difficulty (the lowest value corresponding to the hardest task).

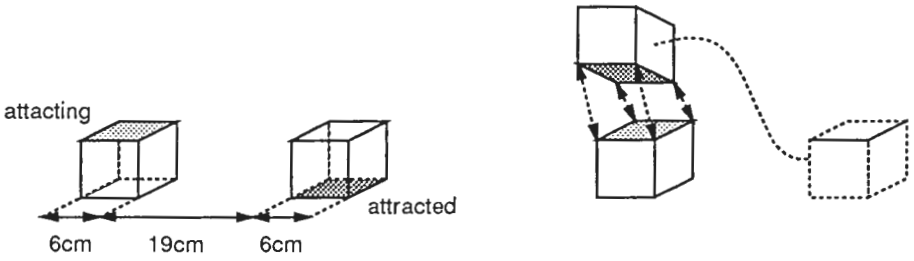

Figure 13: Experiment 2

4.2.2 Results

The table below shows the results of the experiment. Each value of mean and standard deviation is calculated on ten values.

experimentator	difficulty					
	2cm		4cm		6cm	
	without	with	without	with	without	with
no1	4.44 (1.76)	2.55 (0.29)	2.30 (0.61)	1.71 (0.23)	1.71 (0.22)	1.46 (0.18)
no2	8.77 (3.11)	4.09 (1.31)	5.10 (1.42)	3.20 (0.86)	4.10 (0.83)	3.07 (1.16)
no3	14.21 (6.74)	7.87 (3.64)	8.57 (4.49)	4.65 (1.77)	3.92 (1.72)	3.25 (0.88)
no4	13.16 (5.71)	7.37 (2.08)	7.16 (2.94)	5.07 (1.02)	5.87 (1.06)	5.03 (2.12)
no5	20.59 (12.49)	11.05 (6.65)	6.92 (3.83)	4.74 (0.99)	4.85 (1.30)	4.22 (2.91)

The scale is very different from the best manipulator to the worse.

The manipulator no1 was myself. As I had been moving cubes for 2 months before the experiment, as I knew perfectly how to use the "interaction" between objects and finally as I had to test my experiment many times (every time a parameter was changed), the task was very easy for me.

The second manipulator was very skilly at moving cubes in the virtual world. He had done a lot of experiments on the virtual world before being a subject of this one. Consequently his movements were very accurate.

The third and forth manipulators have similar results for a difficulty of 2cm and 4cm. Nevertheless, the time for 6cm is nearly two seconds higher for the forth one, with or without interaction. Their skills for moving cubes was nearly the same. They had already done it, but not as many times as the experimentators 1 and 2. The difference is that the experimentator 4 showed a learning behaviour. With interaction, for the two first trials, he released the cube very carefully, fearing the noise introduced by releasing. Then he understood it was unnecessary for such a difficulty and his behaviour became comparable to the one of the third manipulator (Figure 14 left). Without interaction, though his time went decreasing, he remained timid when releasing, so he could not reach comparable results to the third manipulator (Figure 14 right).

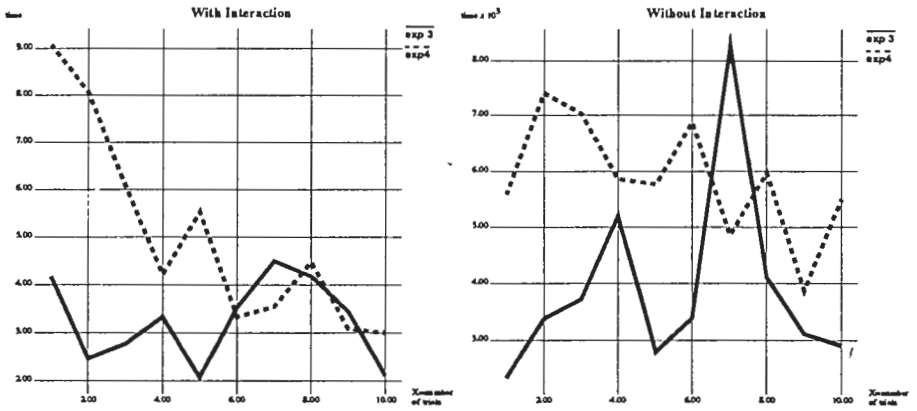

Figure 14: results of the ten trials of manipulators 3 and 4

The fifth manipulator was moving cubes for the second or third time. Consequently the most difficult task was hard to succeed for him. The noise introduced when releasing often obliged him to try again. His results are comparable to those of third and forth manipulator for the other difficulties.

Even with different scales, the behaviours were very similar. Let us analyse the graph of the second experimentator (Figure 15).

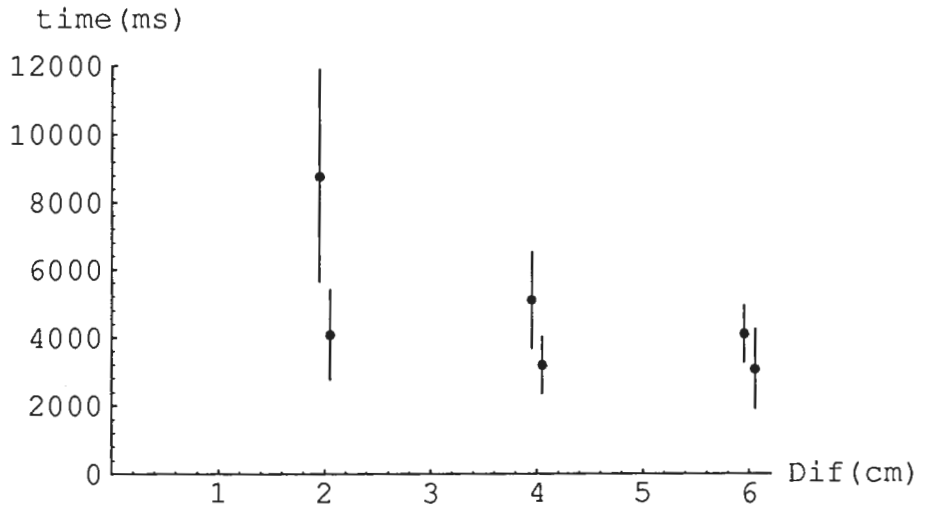

Figure 15: Mean and standard deviation of the time to achieve the task

As only ten values were used, the standard deviation was quite big for each result. Nevertheless the mean was still meaningfull. The general result is that the task was done quicker with interaction and the time spared thanks to this one was increasing with the difficulty of the task.

As we had a common behaviour but very different scales for everybody, we decided to normalise the data by using the following fraction $x_{(\%)} = \frac{t_{with}}{t_{without}}$ and compare the result. This fraction is an indicator to evaluate the interest of the interaction given the following task: move a cube on another with a certain accuracy. We calculated the mean and standard deviation of these indicators for the five experimentators. The results are gathered in the following table (Figure 16)

	difficulty		
	2cm	4cm	6cm
x(%)	53.82 (4.25)	66.12 (7.84)	83.10 (4.85)

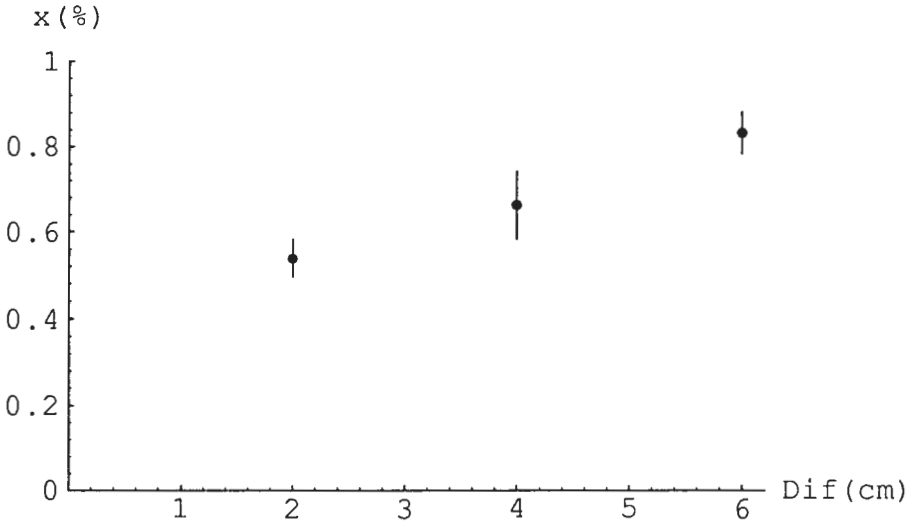

Figure 16: Mean and standard deviation of the indicator x

These results show that the interaction is all the more important than the task to complete is difficult. For the most difficult task, nearly half time was spared thanks to the interaction.

5 Conclusion

For a future work, the tool described in this report could be improved in two ways:

First the choice of the object could be changed. We could use a more complicated distance (thanks to "attracted" and "attracting" faces, the time of computation could still be small enough to have a real time tool). We could also use a potential function including other parameters than the euclidian distance as for exemple the angular distance and the distance between the barycenters.

Second, instead of choosing only one object, a multiple choice should be made. For each "attracted face" an "attracting plane" should be chosen, then the threshold value would eliminate some couples. All the remaining couples would then be used to calculate the movement. An exemple of application that it would enable is to move a cube into a corner.

Acknowledgements

I would like to thank Mr. N. Terashima - President of ATR Communication Systems Research Laboratories, and Mr. K. Habara - Executive Vice President of ATR International, Chairman of the Board of ATR Communication Systems Research Laboratories, for having invited me to come to ATR for five months.

I would like to express my sincere gratitude to Mr. F. Kishino - Head of the Artificial Intelligence Department, who welcomed me in his Department and enabled me to have a broad view of Japanese research on telecommunications thanks to the visit of NTT,NEC and KDD laboratories in Tokyo.

I would like to thank Mr. K. Ishibuchi - Researcher, for his constant support in my research and also all those who accepted to take part to my experiments and who advised me for my work.

Finally, I would like to thank Mr. K. Fujii, Senior Staff of the Planning Division, and all the members of the Planning Division and the Planning Section of ATR Communication Systems Research Laboratories for having organised so perfectly my stay in Japan.

References

[1] D. F. Rogers, J. A. Adams. "Mathematical elements for computer graphics", section 3.8 (1976)

[2] 和田信彦, 鳥山裕史, 田中弘美, 岸野文郎: "凸包を用いた複数レンジデータの統合", 画像の認識・理解シンポジウム (MIRU'92) 講演論文集 I, pp. 373-380 (1992)

[3] F. Kishino "Communication with Realistic Sensations through Integrated Multi-media Environment", Rio Grande Research Corridor

[4] "Graphics Library, Programming Guide version 2.0", Silicon Graphics Computer Systems (1990)

A Appendix I: Inverse a matrix of position

A matrix of position is a 4x4 matrix which contains the three rotations and the three translations needed to define the position of an object in a base. It has the following form:

$$M \stackrel{\text{def}}{=} \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ t_1 & t_2 & t_3 & 1 \end{pmatrix} \stackrel{\text{def}}{=} \begin{pmatrix} \begin{bmatrix} & R & \end{bmatrix} & \begin{bmatrix} 0 \\ \\ 1 \end{bmatrix} \\ \begin{bmatrix} & T & \end{bmatrix} & \end{pmatrix}$$

This is the matrix of transformation from one orthonormal basis to another orthonormal basis. Consequently, the inverse matrix exists: it is the matrix of the opposite transformation.

$$\text{let } M^{-1} \stackrel{\text{def}}{=} \begin{pmatrix} \begin{bmatrix} & R' & \end{bmatrix} & \begin{bmatrix} 0 \\ \\ 1 \end{bmatrix} \\ \begin{bmatrix} & T' & \end{bmatrix} & \end{pmatrix}$$

$$MM^{-1} = I_4 \Leftrightarrow \begin{pmatrix} \begin{bmatrix} & R & \end{bmatrix} & \begin{bmatrix} 0 \\ \\ 1 \end{bmatrix} \\ \begin{bmatrix} & T & \end{bmatrix} & \end{pmatrix} \begin{pmatrix} \begin{bmatrix} & R' & \end{bmatrix} & \begin{bmatrix} 0 \\ \\ 1 \end{bmatrix} \\ \begin{bmatrix} & T' & \end{bmatrix} & \end{pmatrix} = I_4$$

$$\Leftrightarrow \begin{pmatrix} \begin{bmatrix} & RR' & \end{bmatrix} & \begin{bmatrix} 0 \\ \\ 1 \end{bmatrix} \\ \begin{bmatrix} & TR' + T' & \end{bmatrix} & \end{pmatrix} = I_4 \Leftrightarrow \begin{cases} RR' = I_3 \\ TR' + T' = 0 \end{cases}$$

As R is the 3x3 matrix describing the rotations, ${}^tR = R^{-1}$. Consequently we found that:

$$\boxed{M^{-1} = \begin{pmatrix} \begin{bmatrix} & {}^tR & \end{bmatrix} & \begin{bmatrix} 0 \\ \\ 1 \end{bmatrix} \\ \begin{bmatrix} & -T{}^tR & \end{bmatrix} & \end{pmatrix}}$$

B Appendix II: Find the three angles of Euler given a matrix of rotation

In a 3D vector space, any rotation can be decomposed into three angles named Euler angles (Figure 17).

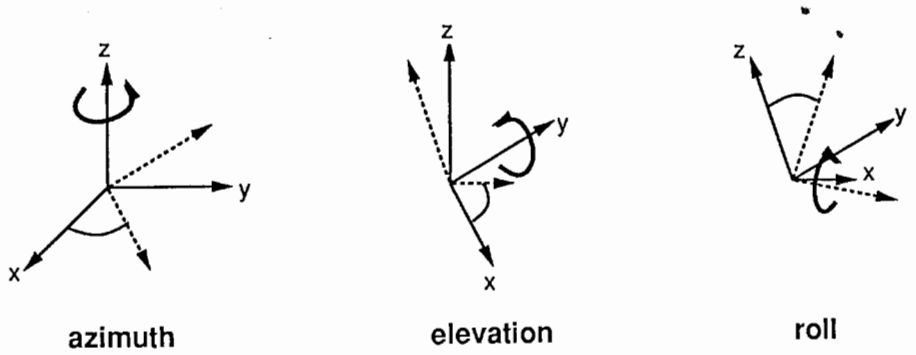

Figure 17: Euler angles

Let ψ, θ and ϕ be these three angles and let R be any matrix of rotation.

$$R \stackrel{\text{def}}{=} \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$$

R can also be written as a product of three matrix of rotation around z, y and x using the three angles of Euler.

$$R \stackrel{\text{def}}{=} \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\phi & sin\phi \\ 0 & -sin\phi & cos\phi \end{pmatrix} \begin{pmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{pmatrix} \begin{pmatrix} cos\psi & sin\psi & 0 \\ -sin\psi & cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\Rightarrow \begin{cases} r_{00} = cos\theta cos\psi \\ r_{01} = cos\theta sin\psi \\ r_{02} = -sin\theta \\ r_{12} = sin\phi cos\theta \\ r_{22} = cos\phi sin\theta \end{cases} \Rightarrow \boxed{\begin{cases} \psi = atan\frac{r_{01}}{r_{00}} \\ \theta = atan\frac{-r_{02}cos\psi}{r_{00}} \\ \phi = atan\frac{r_{12}}{r_{22}} \end{cases}}$$