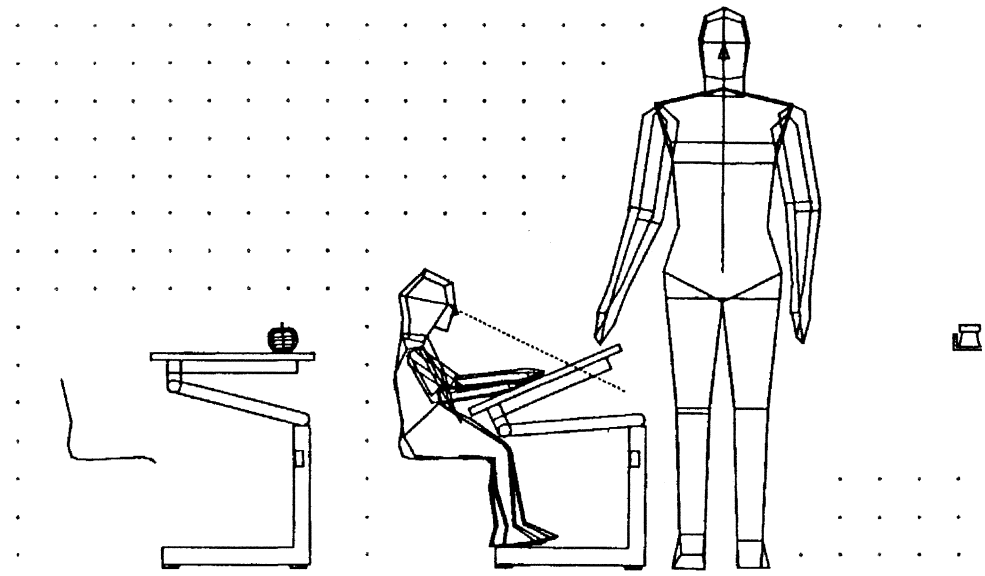
IDE-531 (706)

ADAPS

Manual

Juli 1993



TU Delft
Technische Universiteit Delft

Faculteit Ontwerp, Constructie en Productie
Subfaculteit Industrieel Ontwerpen
Vakgroep Produkt- en Systeemergonomie

Rapportnr	Aantal bladz	Datum
io90 / AE105	160/80	july 1993

Titel

ADAPS - Manual

Auteur(s)

P.N.Hoekstra

Samenwerking met

Bijzonderheden

Samenvatting

ADAPS (Anthropometric Design Assessment Program System) Manual;
PC-version september 1992; MS-DOS; GSSCGI

Trefwoorden

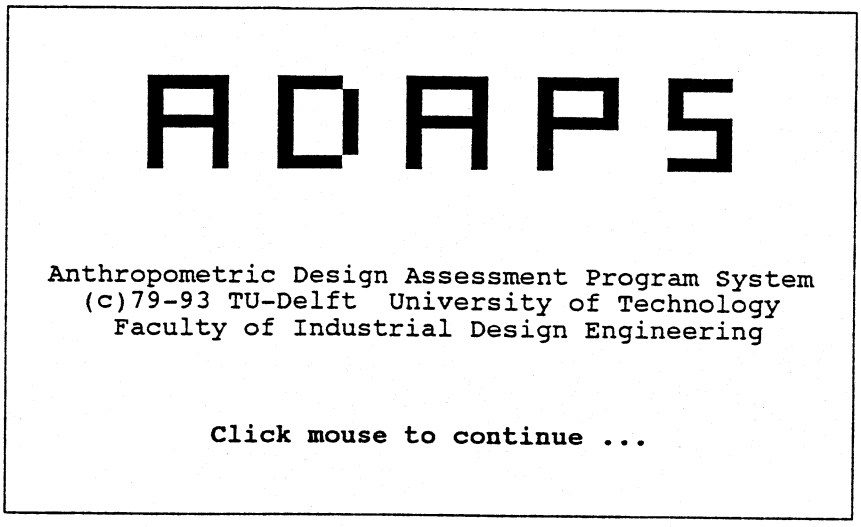
ADAPS, Computer Aided Anthropometric Assessment, Manual

Technische Universiteit Delft
Vakgroep Produktergonomie

Faculteit Industrieel Ontwerpen
Sektie Antropometrische Ergonomie
Jaffalaan 9
2628 BX DELFT

CONTENTS

After starting the ADAPS program, you will get a screen with the next introductory message:



After clicking mouse, you are asked in the next screens to select three specific files:

- a **workspace** file
- a **posture** file and
- a **model** file.

■ Choose a **Workspace file**

The first screen shows a list of available workspace files to choose from.

A workspace file contains the (coded) description of a workspace (see chapter 2.), that you want to visualize and evaluate. It must be an ASCII-file (it should contain readable text), that could have been started off-line, but can be edited on-line (while you are running ADAPS). The name of the workspace file should have the correct extension (.WSP), otherwise it will not be recognised as such by ADAPS and will not be included in the list of workspace files to choose from.

Make your choice by placing the mouse and click-on.

■ Choose a **Posture file**

The next screen prompts you to choose a posture file.

Posture files are binary files, that can contain up to twenty human postures. The ADAPS software includes a number of posture files that contain standard postures for standing and sitting - to help you get started - but you can create your own files and store your own postures while running the program (see later under MENU - CHANGE FILES , page 1-79).

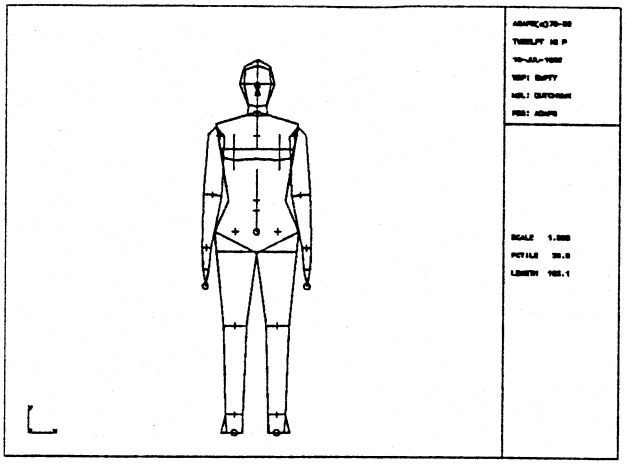
Make your choice by placing the mouse and click-on.

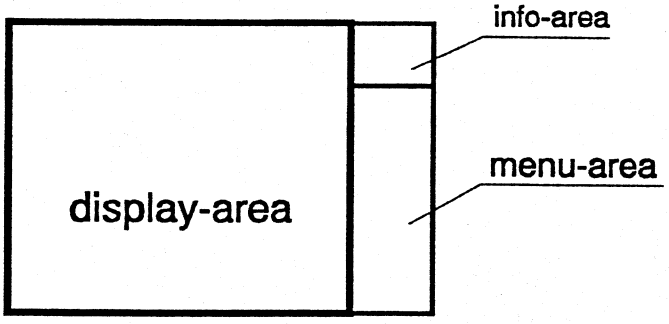
■ Choose a **Model file**

The screen now prompts you to choose a human model to work with.
Again make your choice by placing and clicking the mouse.

The files you have chosen are now connected to the program and the workspace and human model are converted into display-graphics.
You are ready to start using ADAPS.

The screen now should look essentially (we chose an empty workspace) like below:





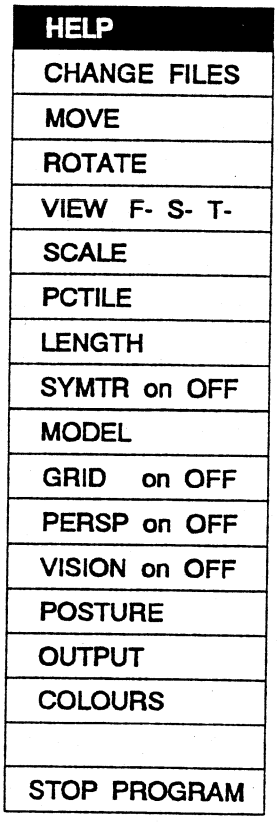
As you can see, it is divided into three areas:

Information area - tells you what files you are using at the moment;

Menu area - lists the menu-items you can choose;

Display area - displays the selected workspace and human model.

Let's start discussing the various menu-items, in a way that seems the most natural.



Menu item **HELP** : - displays general help information;

- lists items that contain their own specific help information.

The Menu

The menu area contains all the available items that you can choose.

Let's begin with the topmost item: click-on over **HELP.**

After this choice you can browse through several pages of general help information; you will also find there the items that contain their own, specific help information. After choosing such an item from the main-menu, you are offered a sub-menu that always contains a new HELP-line for that specific item.

After choosing HELP in the main-menu you can see (e.g.):

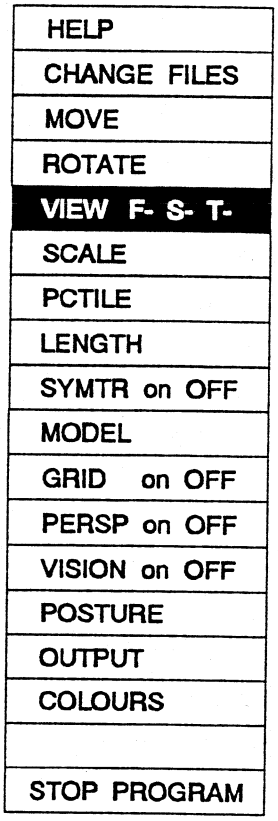
PCTILE Input percentile: .5 < PCTILE < 99.5

or

VISION See HELP in submenu

This means that to get help information about VISION you will first have to choose VISION in the main-menu; after that you can choose the new HELP-line in the sub-menu.

All the help information is stored in the file ADAPS.HLP ; since this file is an ASCII-file, you can make a print and read it outside the ADAPS program too. You can also edit it to your own liking.

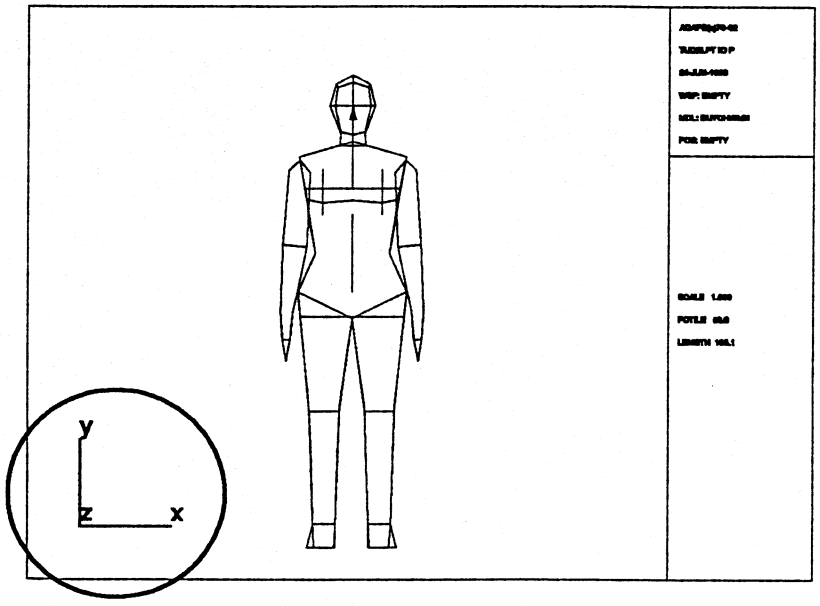


Menu item **VIEW F- S- T-** : choose Front-, Side- or Top-view

VIEW F- S- T-

Let's have a look at option: **VIEW F- S- T-**

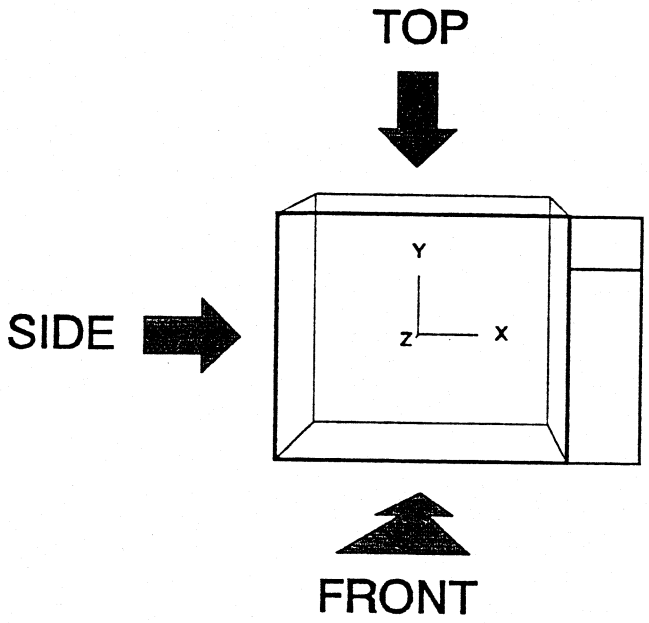
When you start ADAPS, the first display always shows you the default settings for a number of variables. One of these is the setting for VIEW. The default (start) setting is FRONT view:



In the lower left part of the screen you will find an image of the display's co-ordinate axes:

FRONT view: X-axis to the right
Y-axis upwards
Z-axis towards you (implied)

Although the co-ordinate axes are displayed at the lower left of the screen (to get them a bit out of the way), their real origin is at the center of the display area:



As is implied, whenever we have a **FRONT-view**, the Z-axis is directed towards us; we, as viewers, are looking in the opposite direction: the minus Z-direction.

What is actually visible of the scene (workspace and human model) depends on the way the graphics-information is projected onto the screen. What we can see in parallel viewing (see also under MENU item PERSPECTIVE, page 1-35), lies inside a rectangular tube, bounded by the display area's borders.

The tube extends from, in this case, Z = 6 m to minus infinity (we will come back to this when discussing menu-item PERSPECTIVE , page 1-35).

In the picture above we also show what we understand as the

SIDE-view (we look in the X-direction) and the
TOP -view (we look in the minus Y-direction) of the scene.

VIEW F- S- T-

Let's change our FRONT-view into a TOP-view.

Place the cursor over **VIEW F- S- T-** and click mouse.

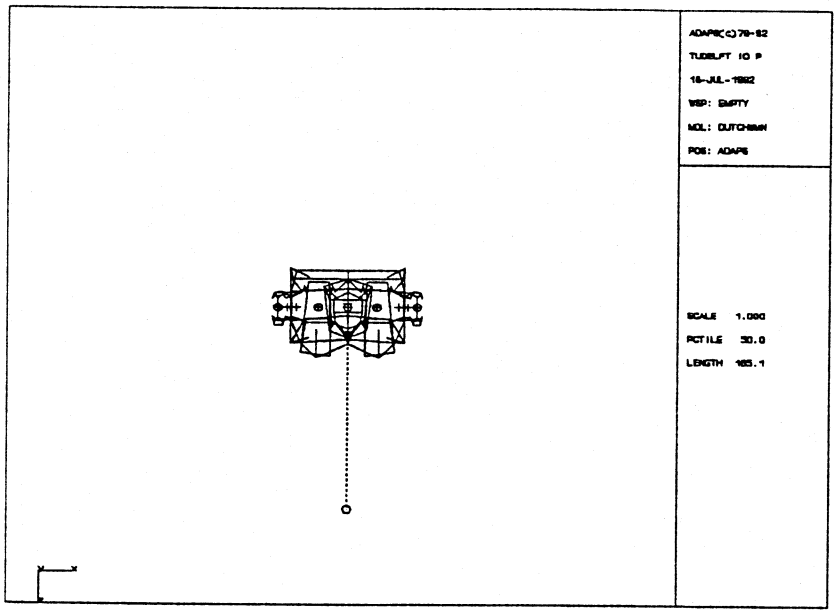
The menu area now shows this submenu:

```
* FRONT view
  SIDE  view
  TOP   view
```

The asterisk (*) tells you what your current view mode is.

Click on over **TOP view.**

The next screen now should look like this:



The display's co-ordinate axes in the lower left now show:

TOP view: X-axis to the right
Y-axis towards you (implied)
Z-axis downwards

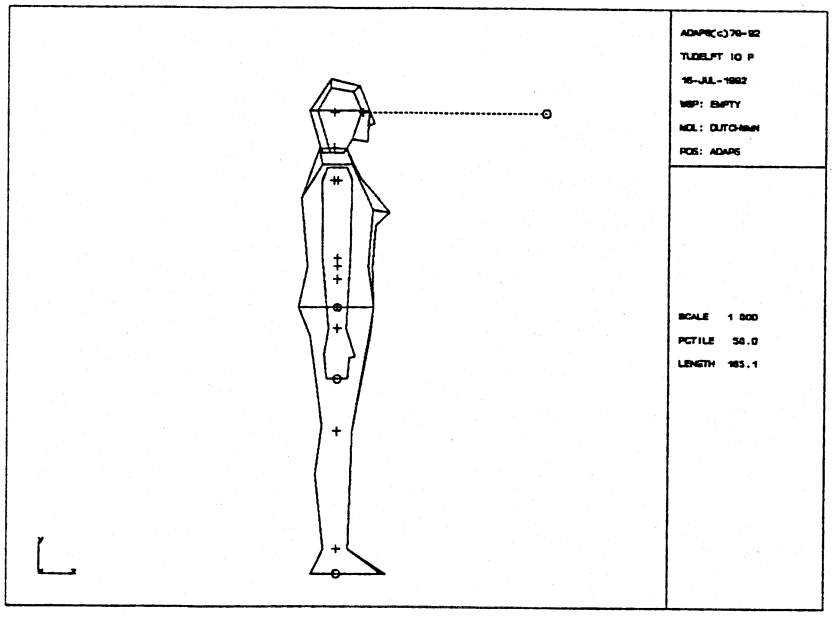
VIEW F- S- T-

The last pre-selected view mode is SIDE-view.

Like before click on over **VIEW F- S- T-**.

The sub-menu now shows:
FRONT view
SIDE view
* TOP view

The asterisk (*) indicates that you have a TOP view.
To get a side view place the cursor over **SIDE view** and click mouse. The screen now should look like this:

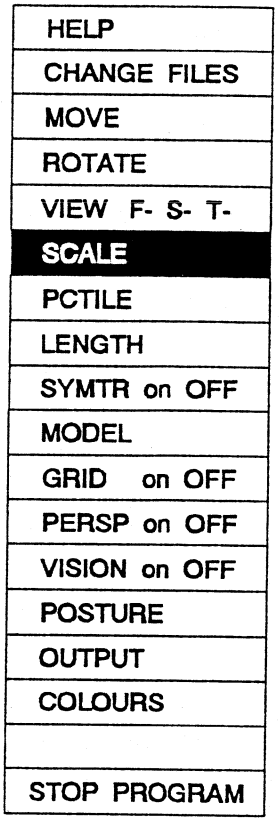


The display's co-ordinate axes in the lower left indicate:

SIDE view: X-axis away from you (implied)
Y-axis upwards
Z-axis to the right

Note that a SIDE view is a view from the left side of the scene (you are looking at the human model's right side).

The three pre-selected view modes FRONT, SIDE and TOP were chosen for easy inspection of the workspace and model. However, you can get **any view** you want by using different settings for ROTATE. We will discuss this option later (page 1-27).

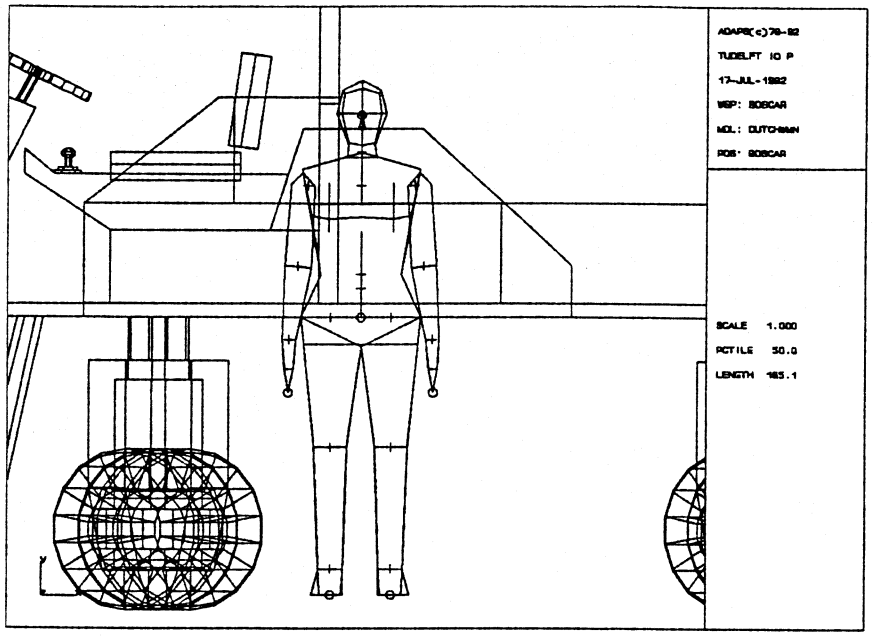


Menu item **SCALE** : scale both workspace and model (zoom in or out)

SCALE

At the start of ADAPS the default setting is: SCALE=1

The first screen of a workplace (e.g. a forest truck) and human model might then look like this:



With SCALE=1 the display-height equals exactly 2 m (200 cm).

The width is a little larger, around 2.16 m. Obviously, the display area is too small to show the workspace in it's entirety. By using the scale option, we can simulate something like zooming in and out.

Let's change the scale. Click-on over **SCALE** . You will find that most menu items remain in place except that at the lower right you are prompted to input a new value for the scale:

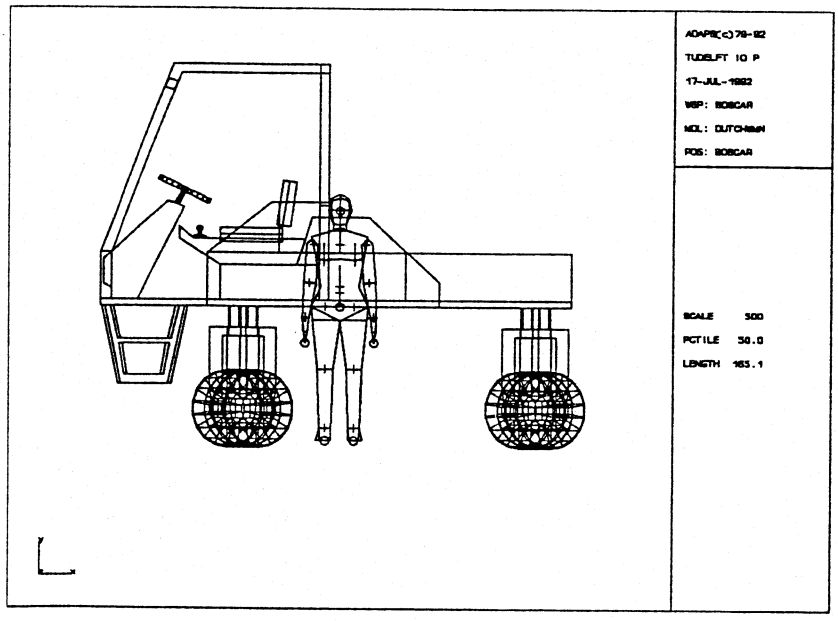
Input Scale
>

You may also have noticed that the mouse cursor has disappeared. You have to enter a value for the new scale.

If you don't want to change anything, just give **<Enter>**. You will find that in that case the scale will stay the same.

Let's make the scale two times smaller (you will then see "twice" as much). The convention in ADAPS for the decimal point is indeed a point, so type on your keyboard **0.5** (or **.5**) and give **<Enter>**.

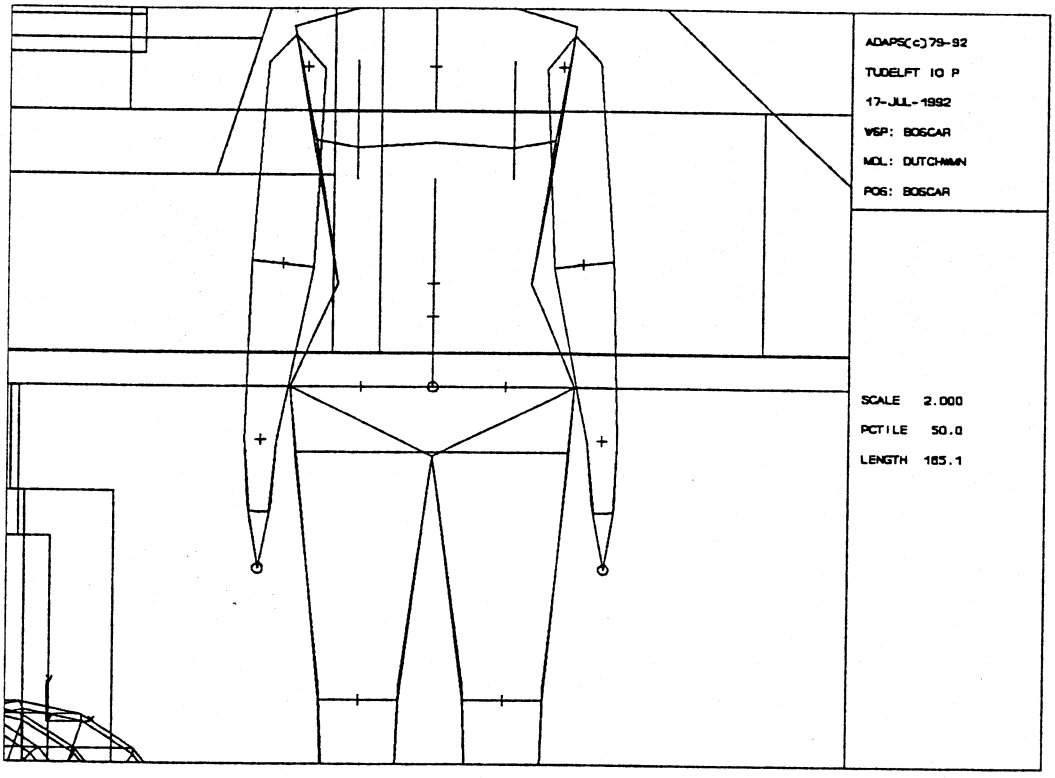
The screen now should look like this:



Note that the value for the scale in the menu area has changed accordingly and now shows .500

SCALE

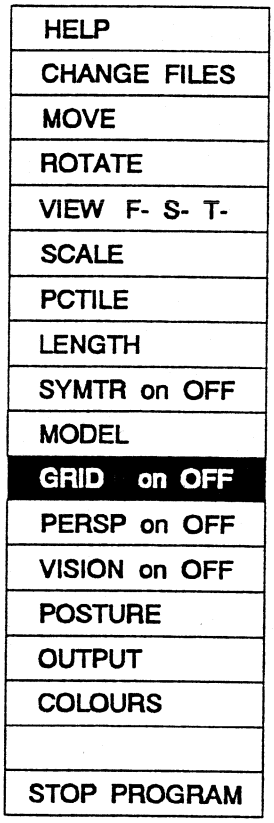
Scale values in ADAPS are **absolute** values and not relative to the latest scale. If you would input a value of 2 for the scale at this moment you would get a screen like this:



So, to conclude this item:

	= 2		100 cm
with	**SCALE** = 1	the **display's height** equals	**200 cm**
	= 0.5		400 cm

You can check these values with the next item we will discuss: GRID a simple but easy measuring tool.

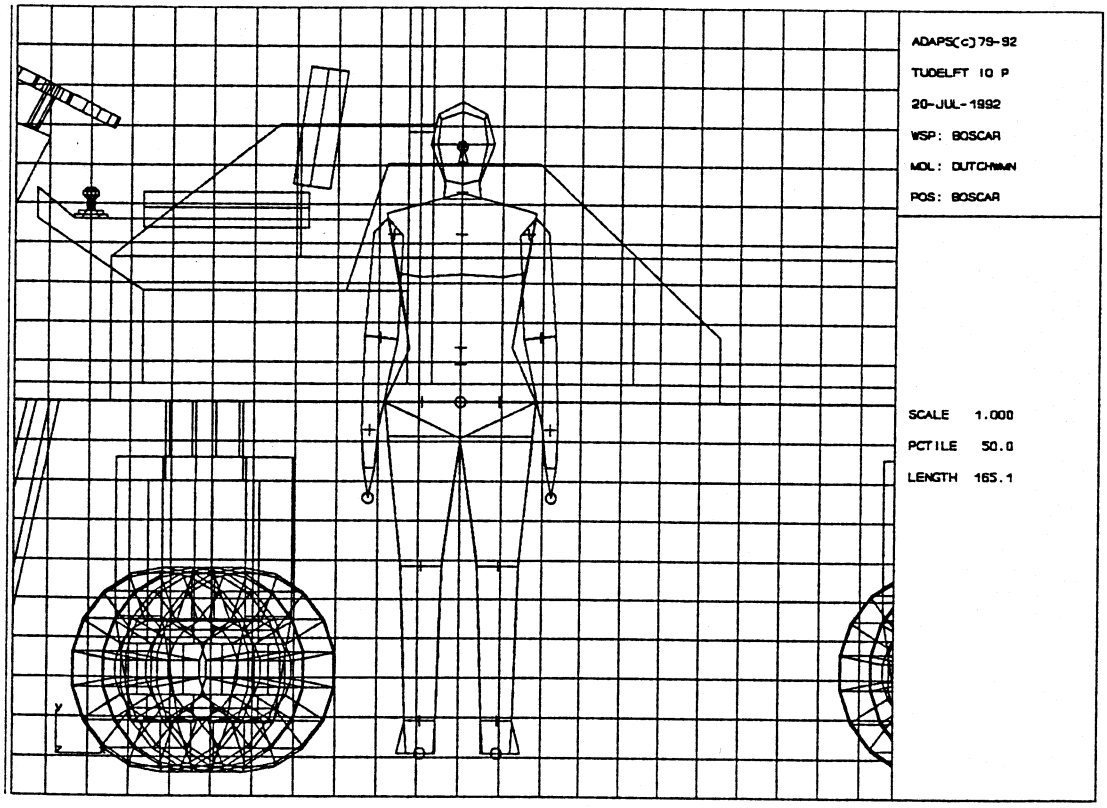


Menu item **GRID** : displays a measuring grid

= 0.5		: 20 x 20 cm
SCALE = 1	**Grid mesh-size**	: **10 x 10 cm**
= 2		: 5 x 5 cm

GRID

Suppose you had just started an ADAPS-session. You now know that in that case the display-variables are set to their default values: SCALE=1, PERSP. on OFF (you have a parallel projection of workspace and human model) and you have a FRONT view. Suppose, like before, you have the truck as workspace and DUTCHWMN as a human model.
Now place the mouse cursor over GRID and click mouse. You should get a display like this:

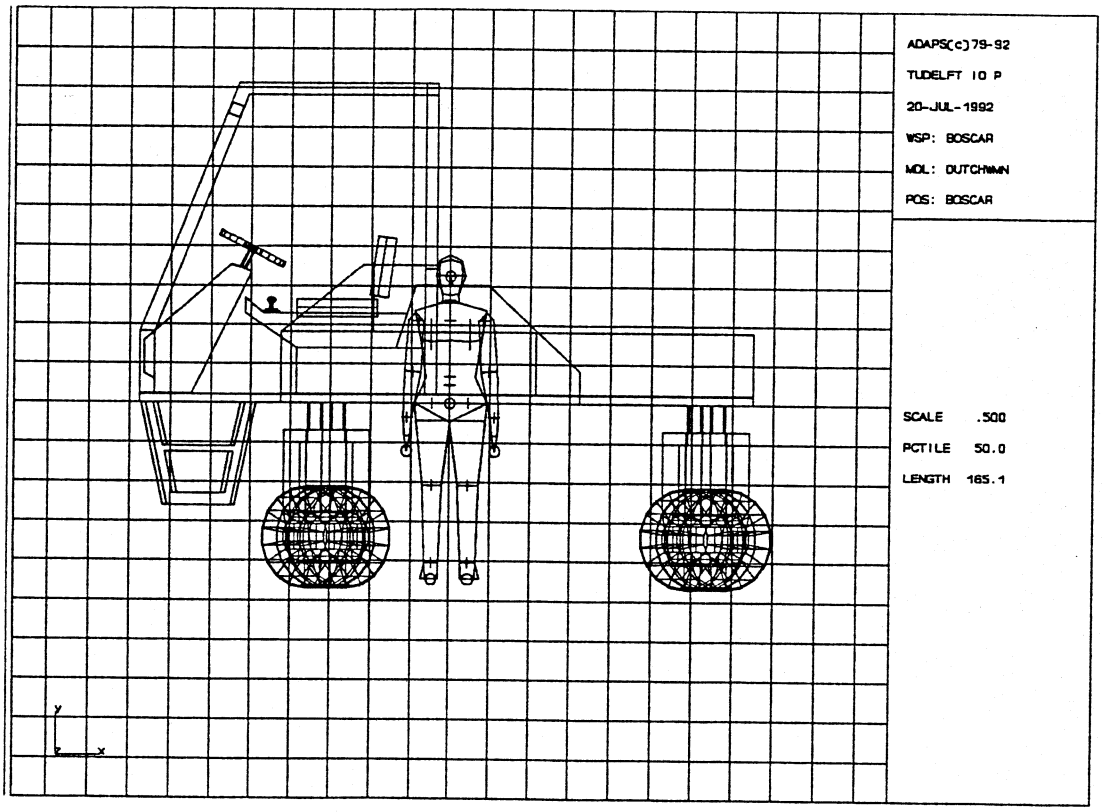


With SCALE=1, this grid's mesh-width measures 10 x 10 cm. Check that the display-height indeed measures 2 m and that the average stature of the model DUTCHWMN, is around 1.65 m.

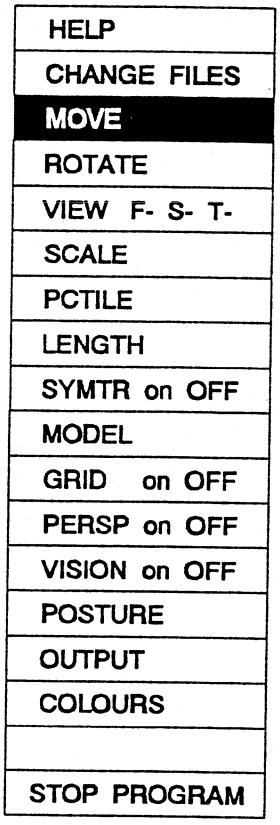
The **grid's display-size** stays the same, whatever the scale; therefore the value for the **mesh-size** will change with the scale:

	= 0.5		: 20 x 20 cm
SCALE	= 1	**Grid mesh-size**	: **10 x 10 cm**
	= 2		: 5 X 5 cm

Check that with SCALE set to 0.5 and GRID to ON you get:



You now can measure the height of the forest truck as around 2.6 m (13 x 20 cm).



Menu item **MOVE** : move both model and workspace along the X-, Y- and Z-axis; unit is centimeter.

MOVE

With menu-option **MOVE** it is possible to move **both** the human model and the workspace in and out of the display area (the workspace might e.g. be too big for this area and you don't want to scale down). MOVE works only along the X-, Y- and Z- axis. Unit is centimeter.

Suppose you started with the default settings for the forest truck and had set the grid on. Now click-on over **MOVE**.
You then get the next sub-menu:

```
=== MOVE ===
select axis
X=          0
Y=          0
Z=          0
END
```

HOME: (0,0,0)

As we can see from the X-, Y- and Z-values: (0,0,0), the scene up till now hasn't been moved (is at "home").
Shift the scene 65 cm to the right while you are in front-view; this means you have to input a new value for the X- co-ordinate.
First select this axis: click-on over **X=** **0** . You now get:

```
=== MOVE ===

X =         0
Y =         0
Z =         0
X - direction
```

and, in the lower right of the menu area:

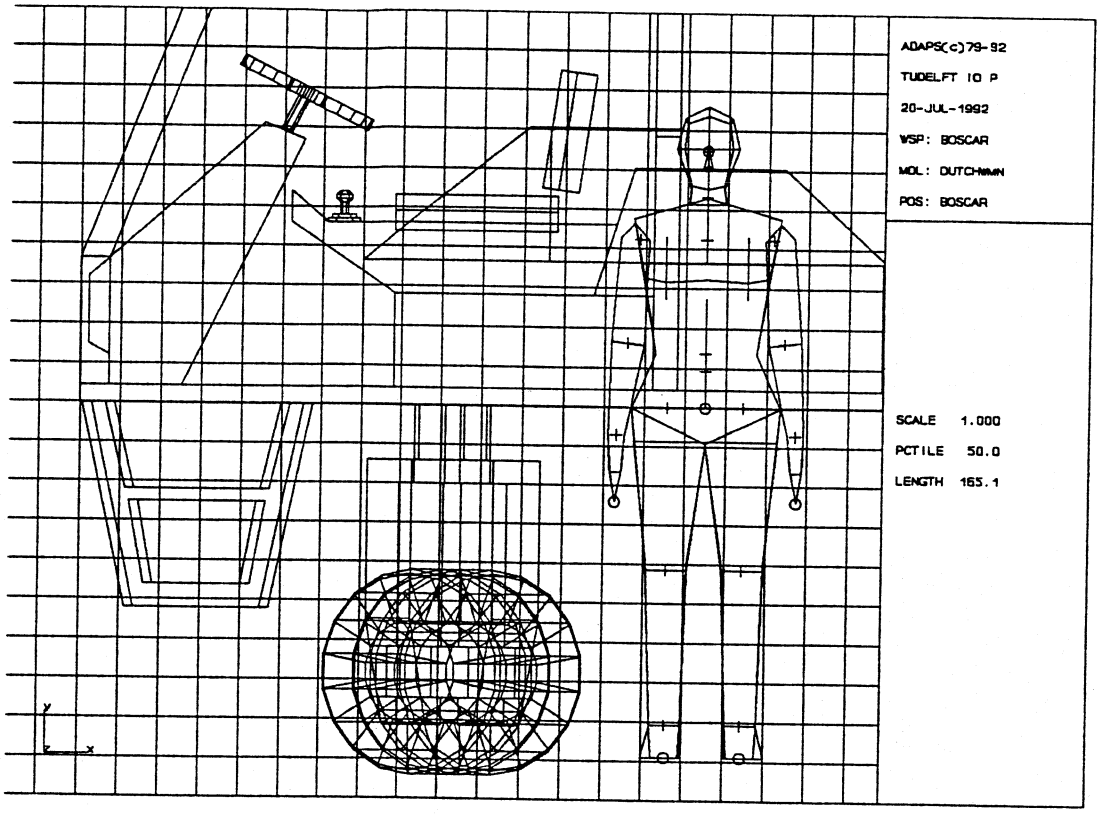
```
Input  X
>
```

Type **65** (or **65.**) on the keyboard and give **<Enter>**.
You will now get the first sub-menu again but with:

```
=== MOVE ===

X =        65
Y =         0
Z =         0
END
```

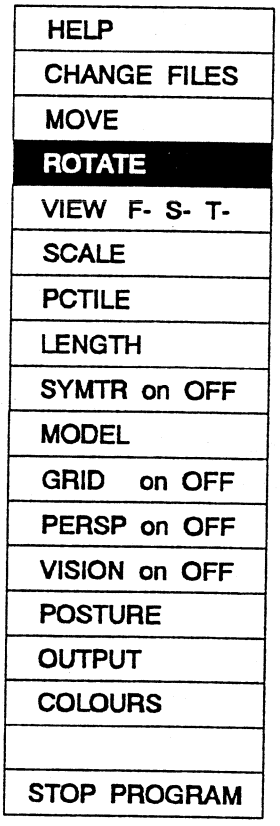
You were returned to the submenu because you might want to change the values for X- and Y- as well. If you would like to see the results, click-on over **END** . You should get a screen like this:



As you can check, **both workspace and model** have been moved the correct amount: the model's base-point now lies 65 cm to the right of the display area's center.

As with SCALE, values for MOVE are **absolute** values and not relative. You can check this: if you next enter a move of X = **-20** the whole scene is shifted so that the model's base-point now ends up 20 cm to the **left** of the display area's center.

If you choose **HOME** in the sub-menu, all values for MOVE are reset to (0,0,0). This implies that the origin of the work-space (defined by the user - see under 2. Workspace), will be put at the center of the display area.

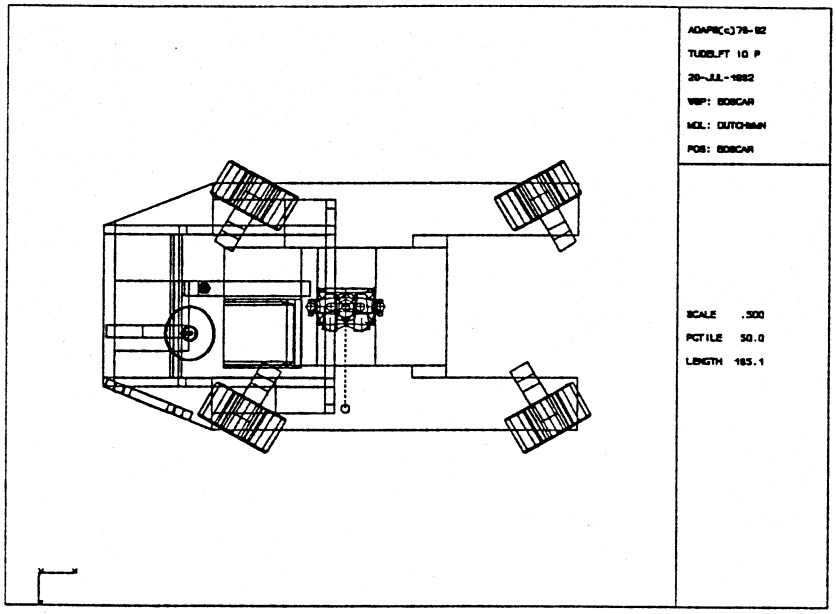


Menu item **ROTATE** : rotate both workspace and model around the X-, Y- and Z-axis **(in this order)** at the center of the display area

ROTATE

With the next menu-option **ROTATE** it is possible to rotate **both workspace and human model** around the X-, Y- and/or Z-axis.

Suppose you had started as before, set the scale to 0.5 and the view to top-view. You then would have a screen like this:



If you now click-on over **ROTATE** you will get a new sub-menu:

```
== ROTATE ==
select axis
X-axis   90
Y-axis    0
Z-axis    0
END

Escape
```

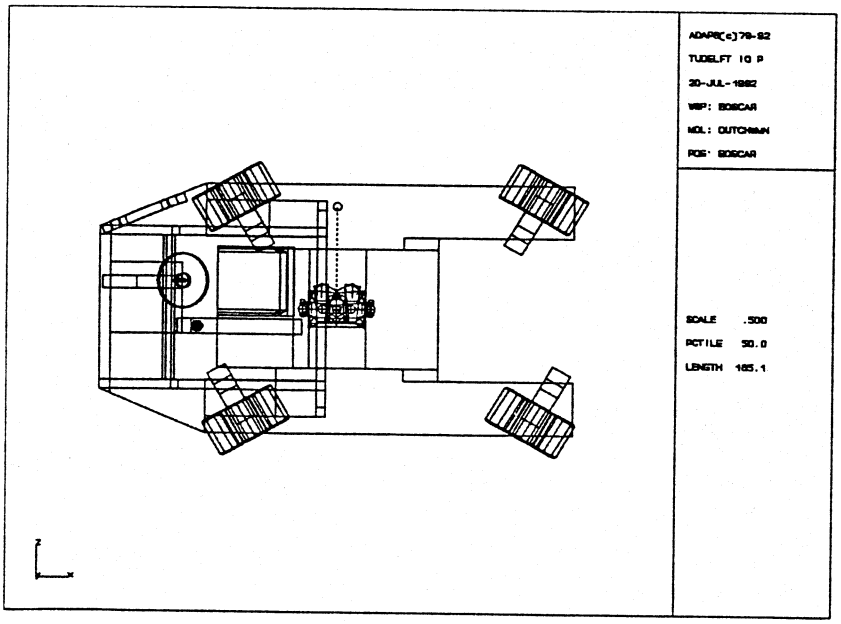
Please note that the sub-menu already shows a value of 90 degrees for the X-rotation. This is because, by choosing TOP view, you implicitly did set the rotation around the X-axis to that value.

Check that a SIDE view would mean: XR (X-rotation) = 0
YR = 90 and
ZR = 0

Just as with MOVE and SCALE, values for ROTATE are **absolute** values and not relative.

You can e.g. get a *bottom* view by: choose **X-rotation,**
input **-90** (with **<Enter>**)
and finish with **END.**

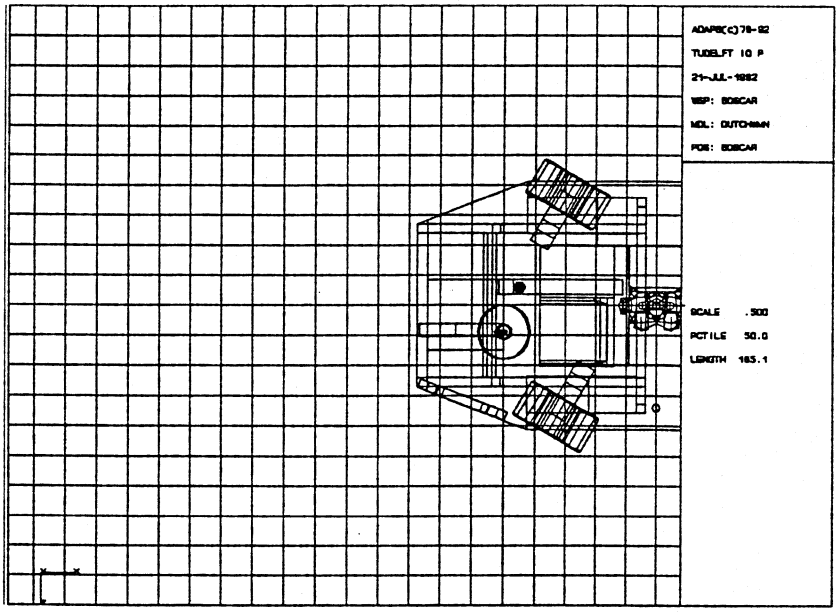
You now should get:



Please note the differences with the previous picture.

ROTATE

Remember that rotation is around the X-, Y- and Z-axes, axes which are situated at the **center** of the display area. In the next example we would like to demonstrate this. We start with the forest truck, scale set to 0.5, we choose a top-view, input 200 cm for an X-move and set the grid on.
We then get a screen like this:



Let's rotate the scene 45 degrees around the Y-axis.
Click-on over **ROTATE** ; you then get the next submenu:

```
== ROTATE ==
select  axis
X-axis    90
Y-axis     0
Z-axis     0
END
```

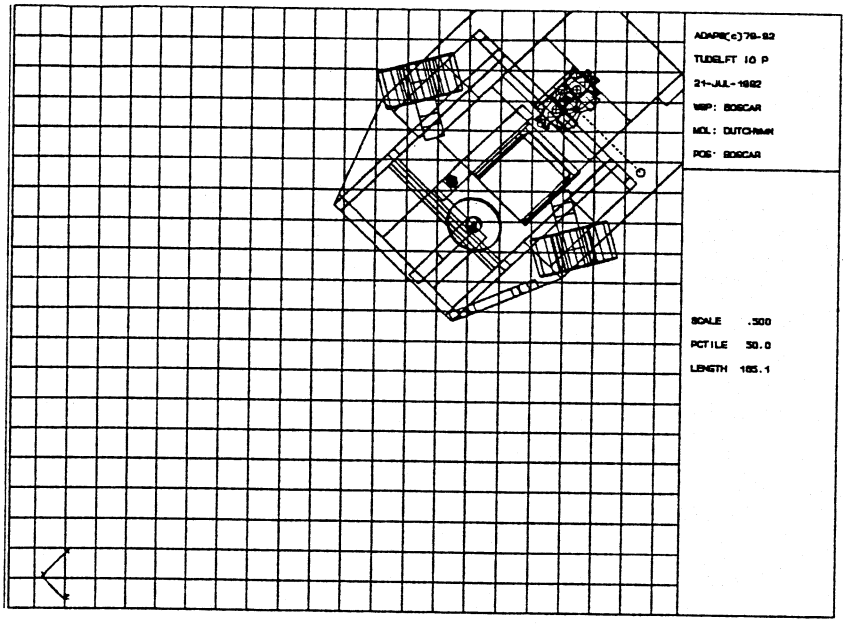
Now click-on over **Y-axis 0** .

You are now prompted to enter a value for the Y-rotation:

Input Y-rot
>

Input **45** , give **<Enter>** and finish by clicking over **END.**

You now get a screen like:



If you compare the last two pictures, you can see that the scene (workspace and human model) has indeed been rotated over 45 degrees around the Y-axis at the centre of the display area.

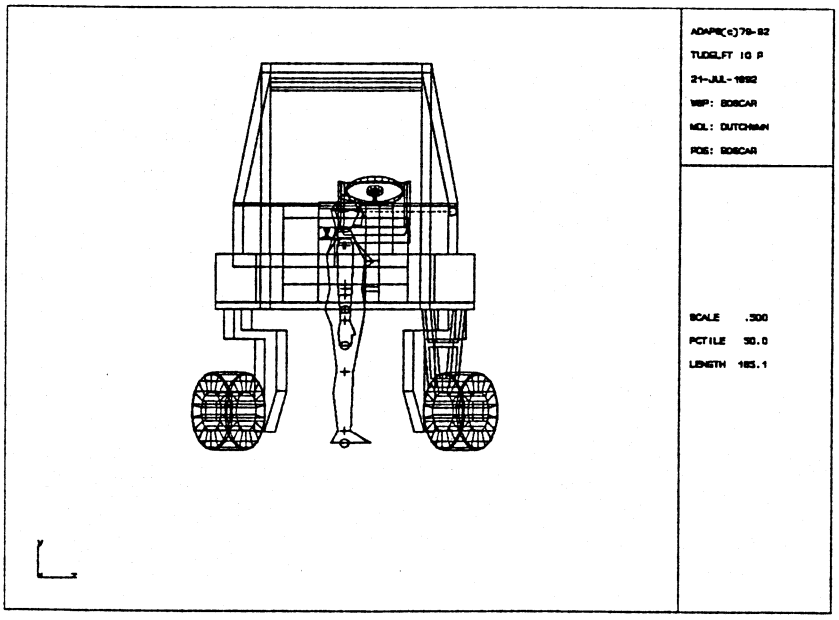
ROTATE

Rotations are not only **absolute**, they are also carried out every time in the same, strict **order**.

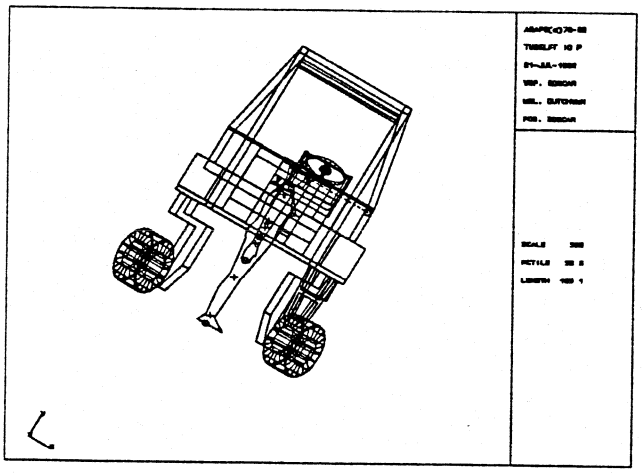
Rotations are always carried out in the same, strict order:

first X-rotation
then Y-rotation around the **new Y-axis**
then Z-rotation around the **latest Z-axis**

This can be contrary to what you normally might expect. Suppose, like before, you had started with the forest truck, set the scale to 0.5 and chosen a side view:

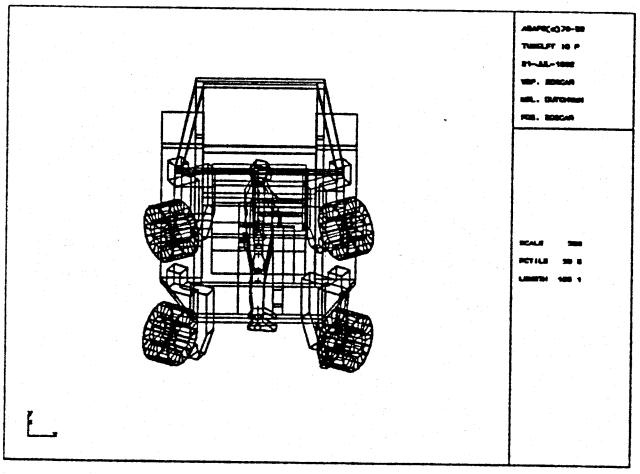


Suppose you would next like to rotate workspace and model over 30 degrees to look like:



You might think, since the X-axis is perpendicular to the screen, that an input **at this time** of 30 degrees for XR might do the trick.

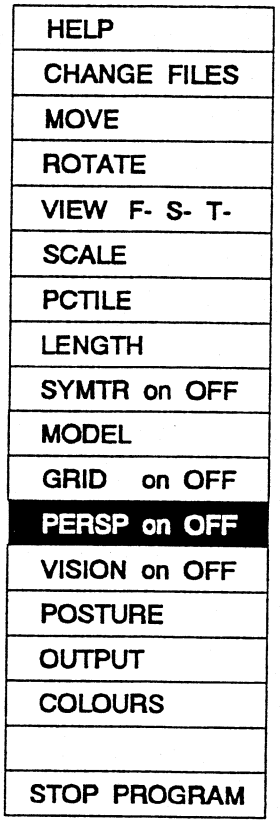
What happens instead is that **first** the scene is rotated around the **X-axis** and only **after that** rotated around the **new Y-axis.**
So, if you have: XR = 30 and YR = 90 (and ZR = 0),
(whatever the order of entering), you will get:



We may change this in a later version of ADAPS to comply with more natural expectations. For the time being you have to get used to thinking "first XR, then YR, then ZR".

You might like to find out by yourself how the one-before-last picture was made (the solution is given at the back of this page).

```
Solution:  XR=  90
           YR=  60
           ZR= -90
```



Menu item **PERSPECTIVE** : display model and workspace in perspective/parallel view

PERSPECTIVE

Up till now, all viewing was done in parallel-projection mode. This is fine for measurement purposes but it doesn't look very natural when you are out of the Front-, Side- or Top-view mode. The next menu-item we will discuss is perspective viewing.

In ADAPS this is a very simple "one-point" perspective where the scene (workspace plus model) is projected from one point onto a projection-plane (your screen), where it's clipped by the borders of the display area (the "window" on the scene). Distance between the point and plane of projection and the opening angle (the angle at wich the point "sees" the borders) are variables that make the scene look more "tele" or more "wide-angle".
Since the display area is fixed, we only have to change the distance between the point and the plane to define the opening angle. For those who are interested in the exact data:

Distance between view-point and plane-of-projection (with scale=1) :	6 m	15 m	30 m
Perspective :	STRONG	MEDIUM	LIGHT
Opening angle (vertical) # :	18° 56'	7° 38'	3° 50'

= 2*atan(screenheight / 2*distance)

In ADAPS you can choose between three, pre-set, distances (three "opening-angles") to get three, slightly different perspective views.

If you have just started ADAPS, you will find in the menu area **PERSP. on OFF** which tells you that the default setting for viewing is parallel-projection (PERSPECTIVE is OFF).
Suppose you had the forest truck again and set scale to 0.5 .

Click-on over **PERSP. on OFF** .
You now get the following sub-menu:

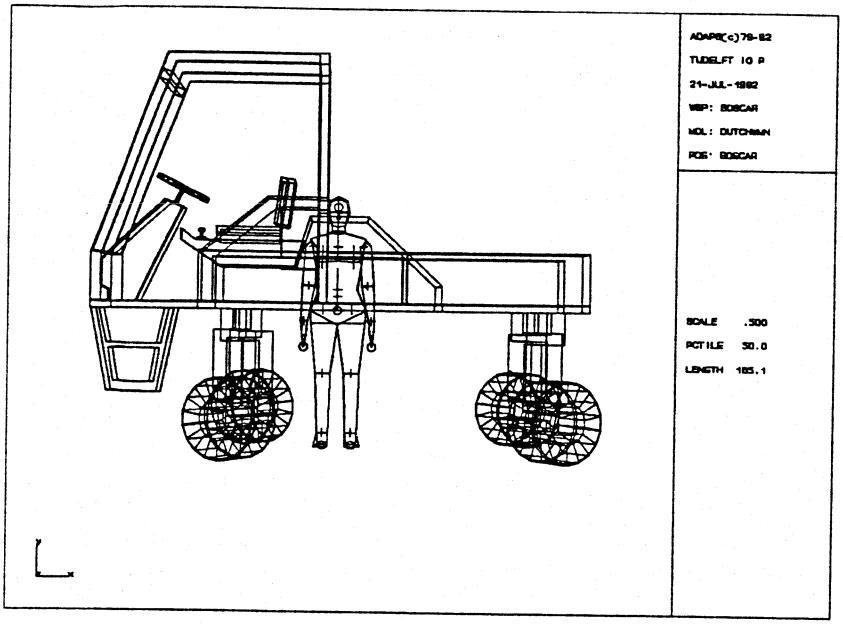
=PERSPECTIVE=

STRONG
MEDIUM
LIGHT
* OFF

Escape

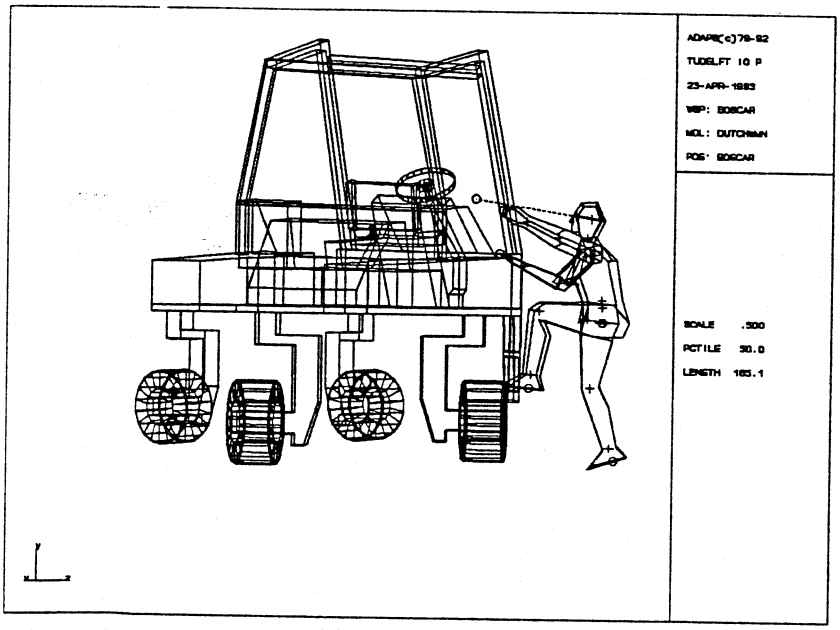
The asterisk (*) tells you what perspective mode you are in.

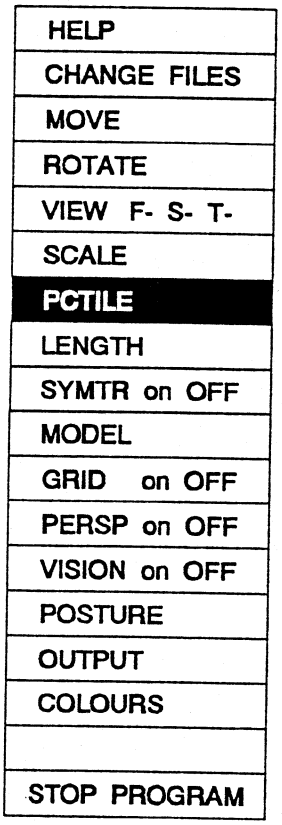
When you now click-on over **STRONG** the scene in the display area will change into:



and the main-menu has returned. It now reads: **PERSP. ON off**

As you can see, perspective viewing, in combination with appropriate values for scene-rotation, may result in a more natural display:





Menu item **PERCENTILE** : enter human model's percentile for length (stature); range: [0.5 -- 99.5]

PERCENTILE

Up till now, the only changes we made were connected with the whole scene (workspace and human model). We will now focus on the human model. One of the options to change the model is its percentile for length (stature); in the menu area diplayed as: **PCTILE 50.0**
This shows us the default value for the human model. If we want to change this, we click-on over **PCTILE 50.0**
We then are prompted in the lower right to enter a value:

Input Pctile
>

We can enter any value for the percentile as long as it's in the range **[0.5 -- 99.5]** . Values are rounded to the first numeral behind the decimal point: so 43.72 becomes 43.7 and 43.76 becomes 43.8 . If we enter values ouside the range (or nonsense like @#$), a message will appear in the lower part of the display area:

Percentile out of range: 0.5 < PCTILE < 99.5

and we are prompted again for a value with:

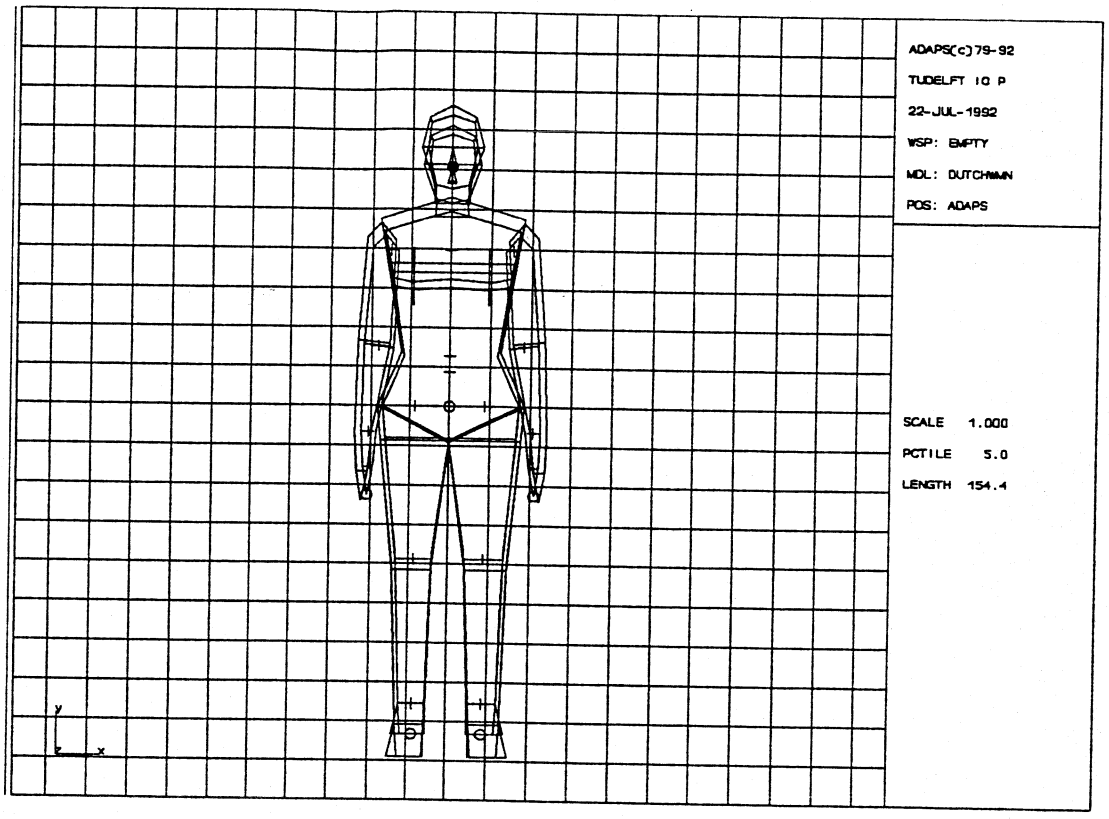
Input Pctile
>

After we enter a valid percentile, the appropriate values for the model-variables (see chapter 3. The Model) are calculated and the scene redrawn. New values for the percentile and corresponding length (stature) of the model are displayed in the menu area.

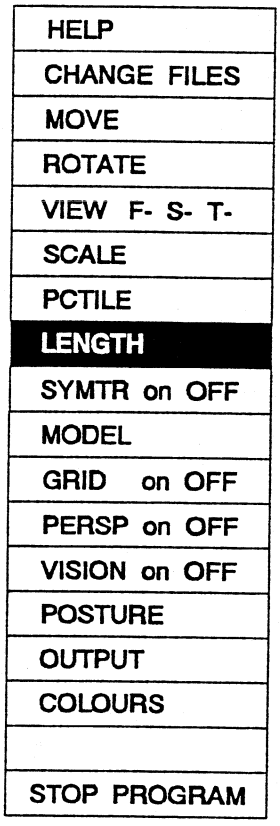
PCTILE and **LENGTH** are connected: if we enter a value for the percentile, the corresponding length is calculated and shown. This is a one-way connection: we shall see later (menu LENGTH, page 1-43) that we can also enter a value for the length or stature of the model, but then no corresponding percentile is calculated.

It is good to note at this point that changing the percentile of an ADAPS model will change the geometry of the **whole** human model, and not only its length: the whole body is scaled upwards or downwards. This means that a P95 model not only has a (correct) greater length (stature), but also (e.g.) a (slightly) bigger head. This presents no problems for variables that correlate well with length; for depth- or weight-related variables however, the user of the ADAPS software must be aware of this aspect.

In the next picture we have put a P5-model over its P50 equivalent (how to do this will discussed under menu-item MODEL ; page 1-97):



Please note that the new model is calculated in such a way that its base-point will stay in place.



Menu item **LENGTH** : input value for the length (stature) of the model; (unit is centimeter).

LENGTH

Besides indirectly (via the percentile-option), we can enter values (unit is **centimeter**) for the length of the model directly by choosing this option by mouse: click-on over **LENGTH** .
We are now prompted to enter a value for the length:

Input Length
>

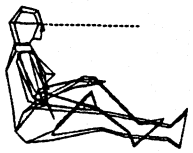
Enter a new value (e.g. **179.6**) via the keyboard and finish with **<Enter>** . Like before with Percentile, a new model is calculated while keeping its base-point in place.
No corresponding percentile is calculated - the menu area will now show: **PCTILE .0**

If we enter nonsense (@#$) after the input-prompt, an error message will appear in the menu area. If we now enter a valid (but also nonsense) input like **400** we will see the next message in the display area:

Length out of range: 50. < LENGTH < 250.

These values are just ad-hoc constraints: length validity is guarded via inputs for a percentile.

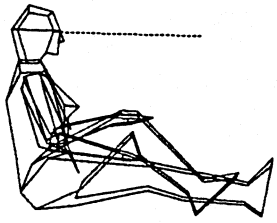
Manipulation



At this point we would like to start showing how to manipulate the anthropometric model into functional postures for a specific work-space.

The ADAPS program offers some nice algorithms as tools to help us, but it still needs the user, someone experienced in assessing human workspace-interaction, to use them in a sensible way.
We should always keep in mind that there exists a difference (to put it mildly), between the anthropometric model (and its assumptions) and a real human being. Mind for instance, what Levis et al. (1980) say about "Manikin mania and anthropomorphism":

"Providing a plastic manikin and a set of design targets for limb angles, which are "comfortable", generates a recipe for disaster. (...) Ergonomists can be tempted into this trap when using a more complex and more sophisticated man model."



The anthropometric model (or mannekijn, to use an old dutch word for 'puppet' or 'little human') can be manipulated in three different ways:

- A. by reading **postures**, stored in so-called posture files (pre-stored by us or by the user).
- B. by using - a **displacement-algorithm** for the whole model;
  - **reach-algorithms** for the arms and legs;
  - a **direction of view algorithm** for the line-of-sight.
- C. by entering values for the **joint-angles** of a chosen body-member (link).

A. Read postures from Posture-files

We will discuss this later under item POSTURE (page 1-83)

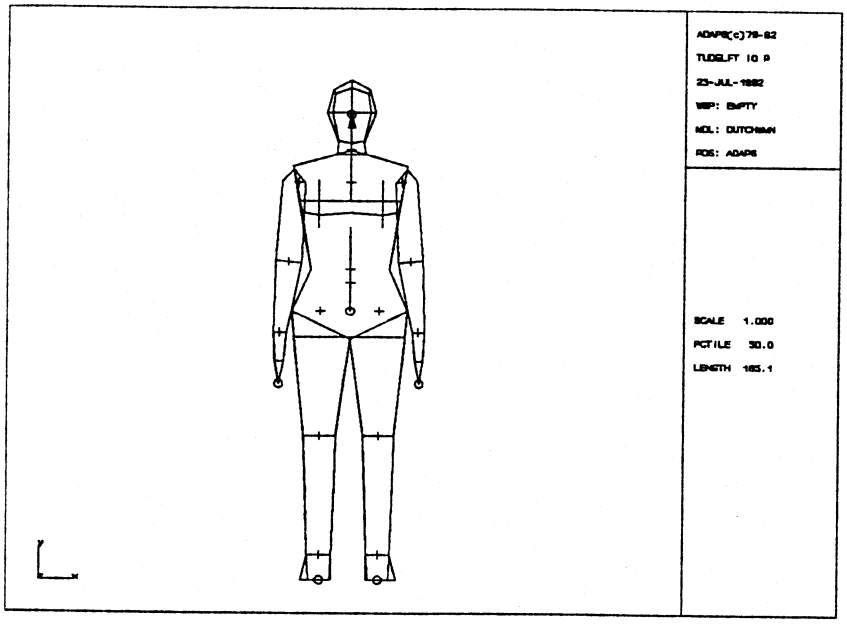
B. Manipulate model using algorithms

These algorithms allow us to

- move the model around the workspace
- position the arms and legs
- tell the model where to look

in a very easy way: we just use the mouse cursor to "tell" where it (they) should go to.

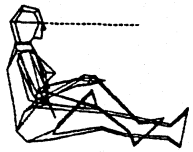
If we take a look at the next picture:



we see at the base-point (the place between the two hip-points) and at the ends of the arms and legs a **special marker.** On our **screen** they appear as tiny **rectangles,** in the picture above they are little circles. If we activate them, we can activate an appropriate algorithm.

When a marker is activated it will change from a rectangle into an asterisk (*).

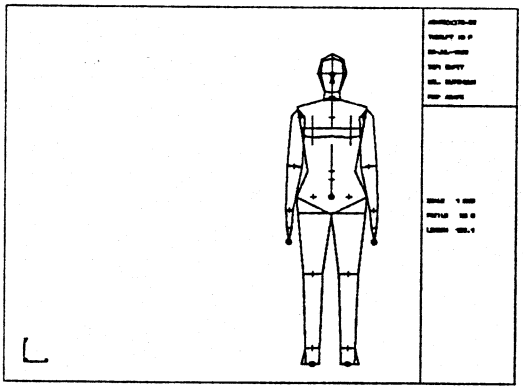
Manipulation
Move base-point



If we want to **move the model** around in the workspace, we have to activate the **base-point** while in **front-** , **side-** or **top-view.**
After we have done this, we see a submenu appear for the base-point (just skip this for the moment) and in the lower part of the display area, there is the next message:

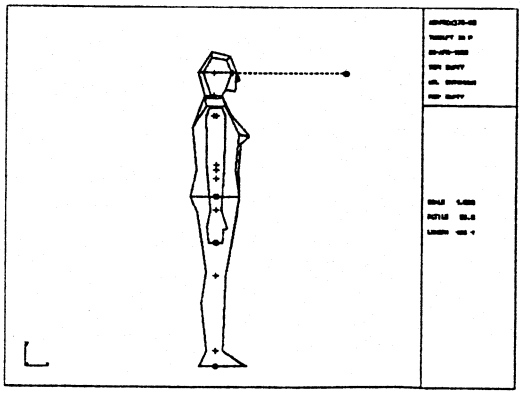
Indicate new position or select menu

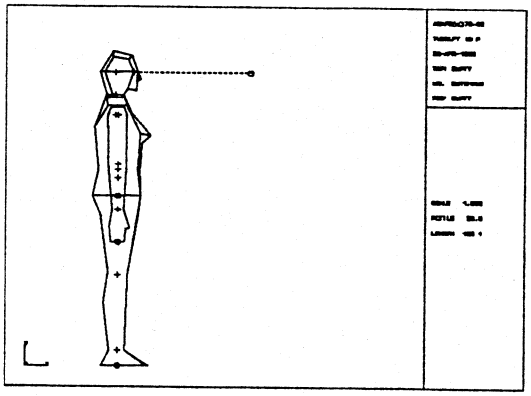
We place the cursor a bit to the right and click-on.
We then will get a screen like:



As we can check, the model has moved, with its base-point at the place we indicated.

If we now choose the side view via **VIEW F- S- T-** , we will get a picture as below left. Activate the base-point again. We get the same message as before: "**Indicate new position ...**".
Place the cursor a bit to the left and click mouse. The next screen will be like below right:





Manipulation
Move base-point

In this way, by combining **mouse-positioning of the base-point** in two of the three pre-set views **front, side** or **top** , we can place the model anywhere we like in the workspace.

Let's return now to the sub-menu. Whenever we activate the base-point (even if we are not in front-, side- or top-view), we get the following sub-menu:

```
=CHANGE POS.=
BASE-POINT
FI    =  -90
THETA=   -90
PSI   =    0
X     =    x
Y     =    y
Z     =    z
END
```

Skip FI, THETA and PSI for the time being. They are used for oriënting the whole body, wich we will discuss later (see "joint-angles", page 1-59).

The next three lines however, can be used to indicate that we want to input exact values for the place of the base-point. If we just had started, the base-point is at the origin so x, y and z = 0. **Units** are, like before, in **centimeters**. Values are **absolute** and not relative.

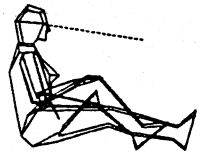
Suppose we started with the scene of the forest truck and had set the grid on. Let's shift the model, while we are in front-view, for instance 40 cm to the right and 20 cm up:

Activate the base-point.
In the sub-menu click on **X = 0**

We are then prompted in the lower right:

Input X-move
>

We input **40** via the keyboard and give **<Enter>**

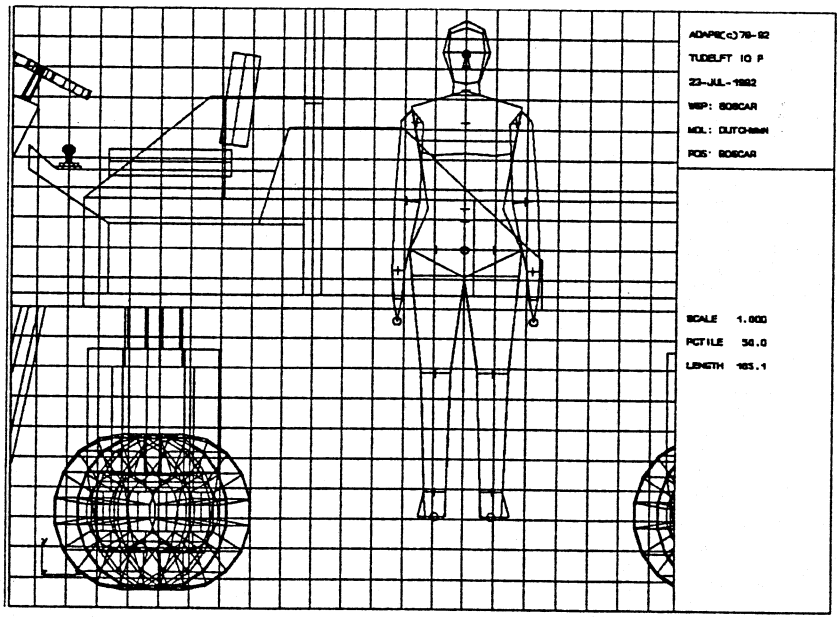


We can see that the input is echoed in the sub-menu but the display doesn't change as yet. Repeat the above for the Y-value:

- click-on over **Y = 0** ;
- enter **20** via the keyboard and give **<Enter>**

When satisfied, we finish with **END** .

Our next screen will be like:



Check that only the human model has moved.

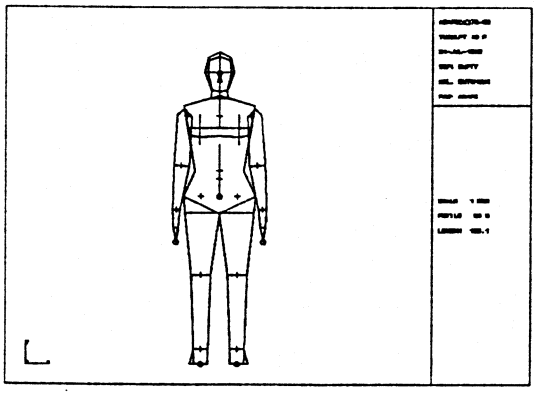
Manipulation
Reach

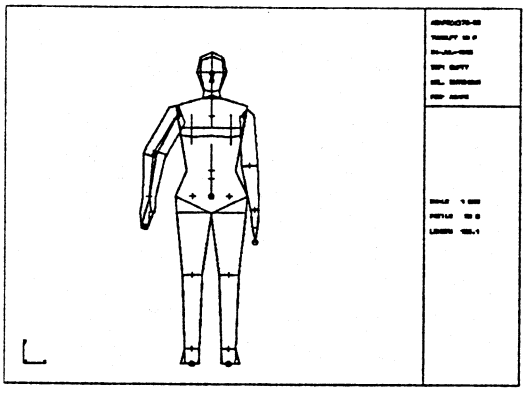
Whenever we activate one of the markers at the end of the arms or legs while we are in **front-**, **side-** or **top-**view or while we are in the field-of-view mode (VISION is ON), a **reach-algorithm** can be activated.
By mouse positioning we can then indicate, where we want an arm or leg to go to. Shoulder/(Hip) - points remain fixed.
Joint-angles of the relevant hand/foot are reset to default
(see later page 1-59).

Let's activate the marker at the end of the right arm.
We get the same message as before:

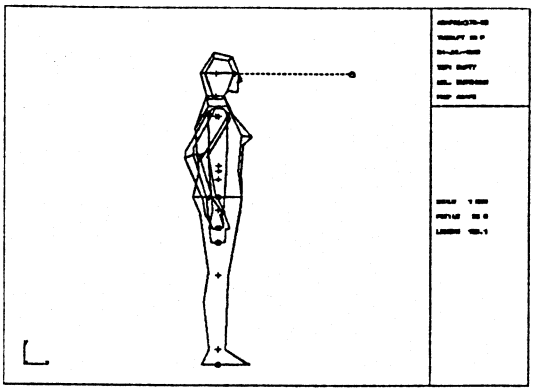
Indicate new position or select menu

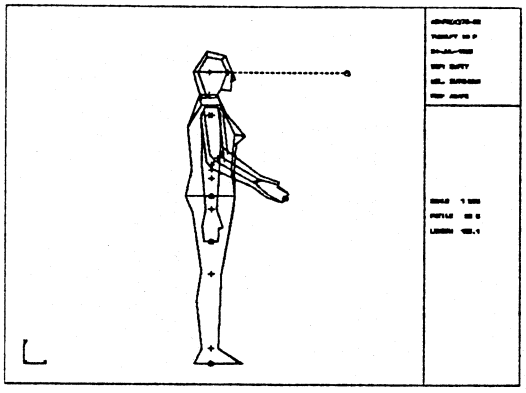
We skip the menu for the moment and just indicate by mouse where the new end-point for the activated arm should be:

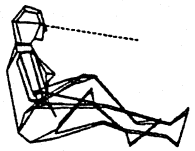




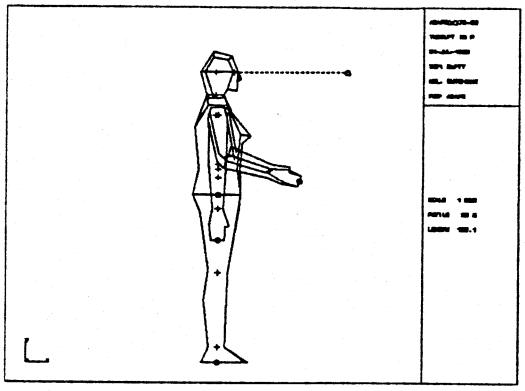
Go to the side-view and repeat: activate the end marker of the right arm and indicate by cursor where it should go:

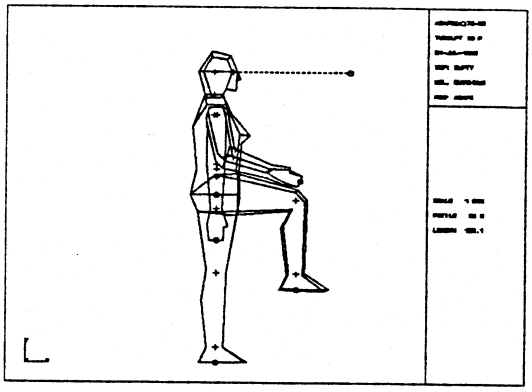




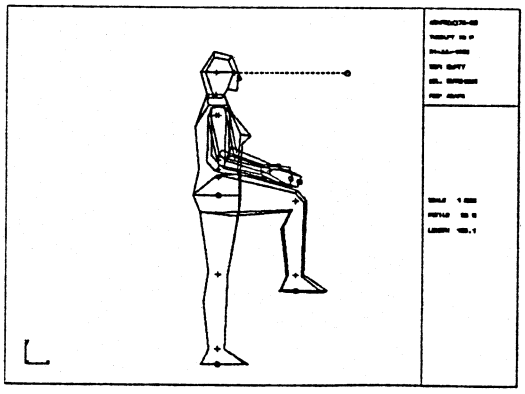


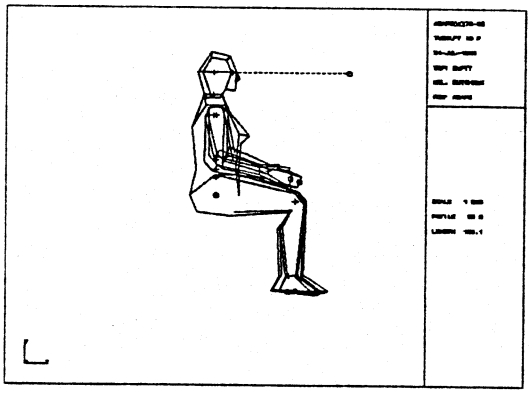
As you can imagine, mouse-positioning results in only **two** new co-ordinates - the "third" (the one perpendicular to the screen) is taken to be the "third" co-ordinate of the relevant body-member. In this way it's very easy to switch between views and continue a reach to get e.g. the model from its starting posture into a sitting posture: with the last model we activate the marker at the end of the right leg and indicate by mouse where it should go:





We repeat this for the left arm and leg:

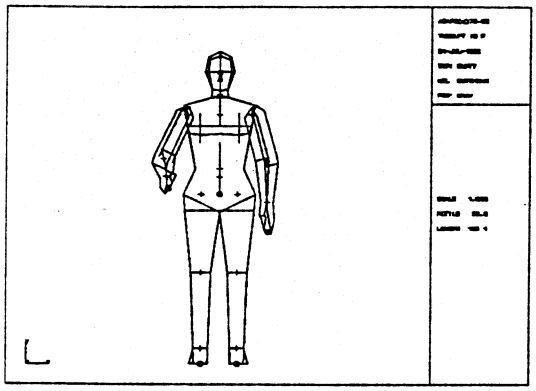


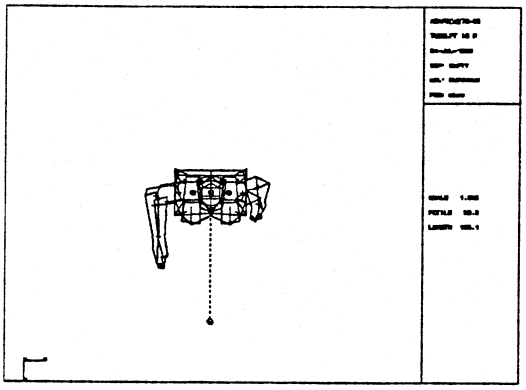


Reach-algorithms for the arms (legs) can be turned on by activating the marker at the end of the arm (leg), while we are in one of the three pre-set view-modes: Front-, Side- or Top-view -or- while we are in the field-of-view mode (VISION ON off).

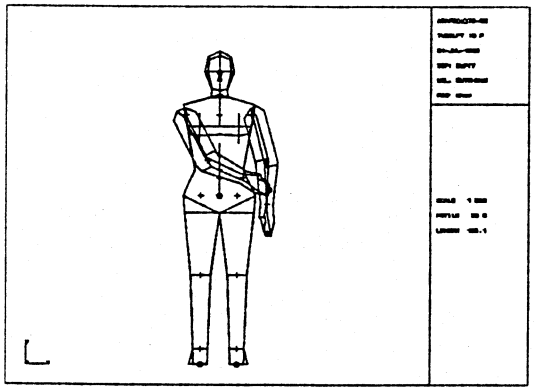
Manipulation
Reach

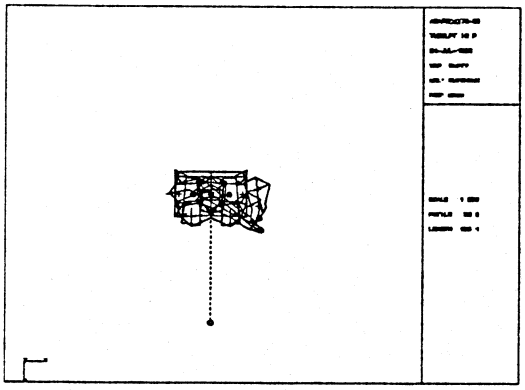
Reach algorithms for the arms and legs can be nice tools to put the human model into a specific posture. They don't know however, **what** they are reaching for (or **why** at all): is it a control or a hot cup of coffee ? They don't recognize objects as possible obstacles (the model isn't "aware" of its own parts). Besides, joint-angles are only checked for their individual range. We could e.g. have the following situation in front and top view:



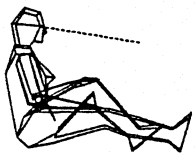


If we now activate, in top view, the model's right arm and told it to go just in front of the left arm, we would get a screen as below right:





This looks all right in front view but combined with the top view you can see that the model sticks her arm straight through her body.



This kind of problem can only be overcome by inputting individual joint-angles for the relevant body-members (page 1-59).

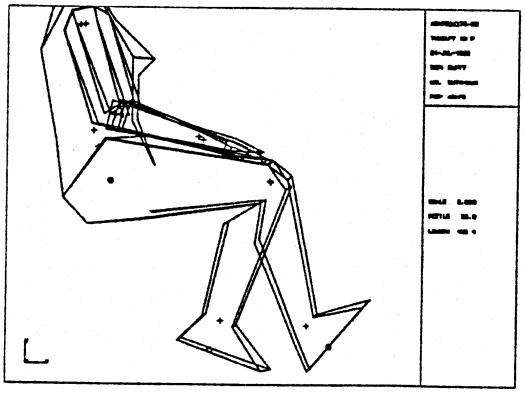
The reach algorithms are not accessible when we are not in the above mentioned pre-set views or in the field-of-view mode. If not, when we then activate an arm or leg, the message in the display area just reads: **Select menu** . The sub-menu can now be used to input individual joint-angles for, in this case, a hand or a foot:

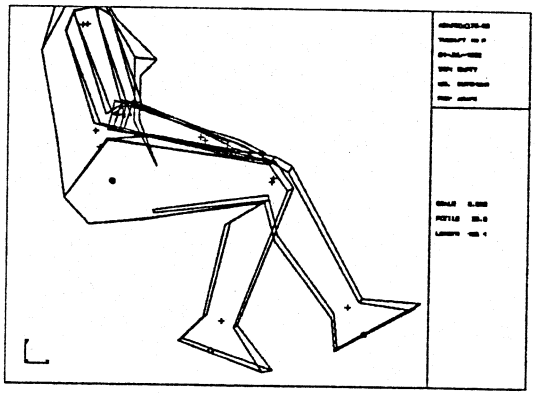
```
=CHANGE POS.=
RIGHT HAND
FI    =  fi
THETA=  th
PSI   =  ps
END
```

We shall discuss this later (page 1-59).

Remember what we mentioned earlier about the reach algorithms: whenever we use them, the **individual joint-angles** for the relevant **hand** or **foot** are **reset** to their default values!

If e.g. we already had entered individual angles for the right foot as in the left picture and - after that - used the mouse again to reposition the right leg, we would end up with a situation as in the right picture:





It's always best to first use the reach algorithms as a rough guide and only after that input values for the hands and feet.

Manipulation
Line-of-sight

The last manipulation-algorithm in ADAPS, concerns the line-of-sight. At the end of the line-of-sight, you find the same marker that was used for the base-point and for the ends of the arms and legs. This implies that when you activate the line-of-sight in a like manner as before (click-on over the marker), you can indicate by mouse a point of interest where you want the model "to look at".

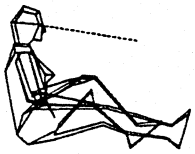
You can do this:

- when you are in the normal preset view-modes: Front-, Side- or Top-view;
- when you are 'seeing what the model sees' (**VISION** is **ON** , see later, page 1-69).

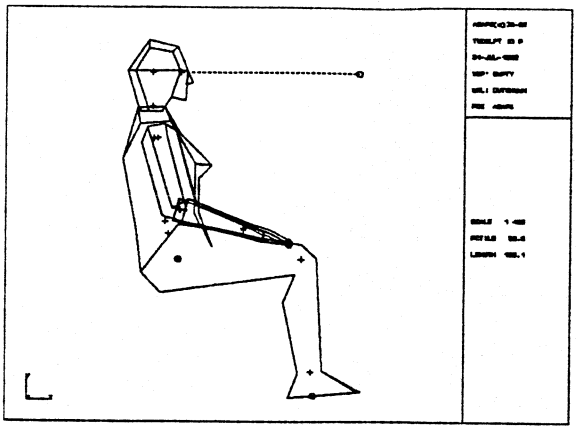
After indicating where the model should look, the line-of-sight algorithm calculates a new head and neck posture.

As you can imagine, this possibility, combined with the VISION option (that displays the model's field-of-view), can be used as a nice tool for visual inspection of the workspace "as if you were the chosen model yourself".

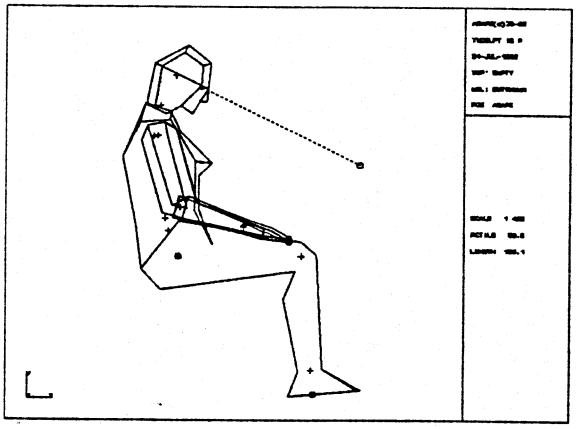
When the algorithm is used, individual "joint"-angles of the line-of-sight (you can input them too: see later Joint-angles, page 1-59), are **reset** to their default values (looking straight ahead).



Suppose you had the scene below:



Now click on the line-of-sight marker and indicate where you want the model to look at (place cursor and click mouse). You then can get a scene like:

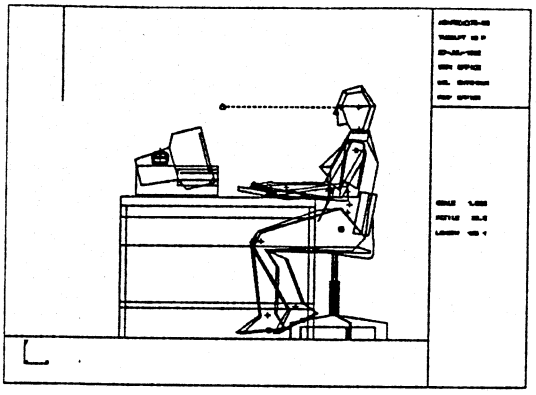


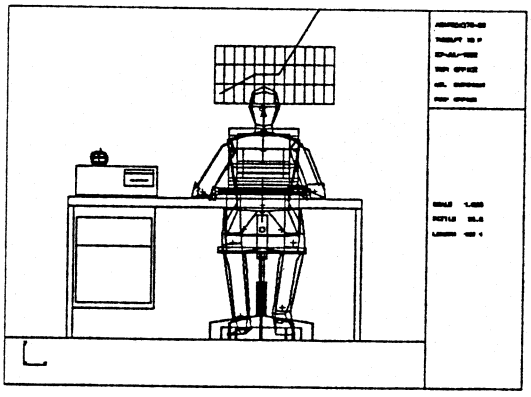
When you combine this "look at" in another pre-set view you can accurately focus the model's line-of-sight to any point of interest in the workspace (within the individual range of joint-angles for the line-of-sight, the head and the neck).

Note: the above is accomplished by storing the coördinates of this 'point of interest'. However, the **first** time that the algorithm is used, the program uses the origin (0,0,0) to get the 3rd co-ordinate (the one perpendicular to the screen), which can result in an unexpected posture (e.g. looking a bit sideways or a bit down). The next time, in a different view, your point of interest is set properly.

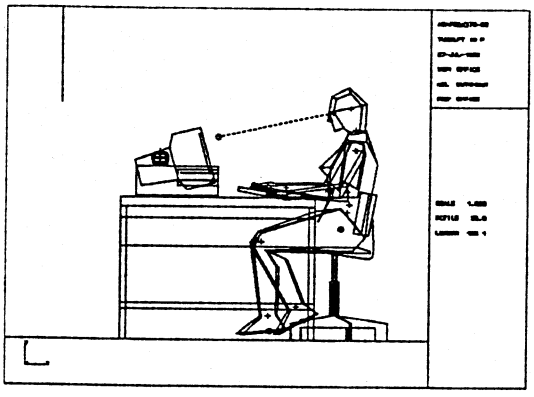
Manipulation
Line-of-sight

If you had e.g. the office scene as below, in side- and front-view (we are looking from behind the model):

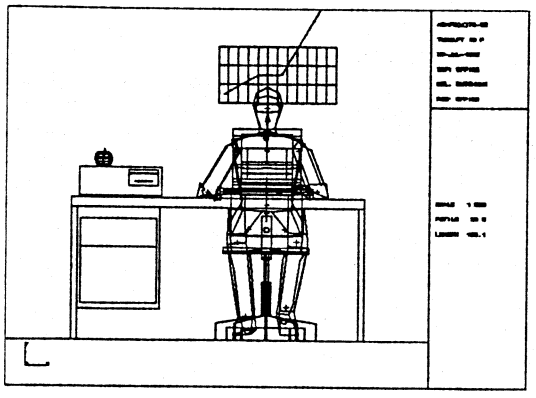


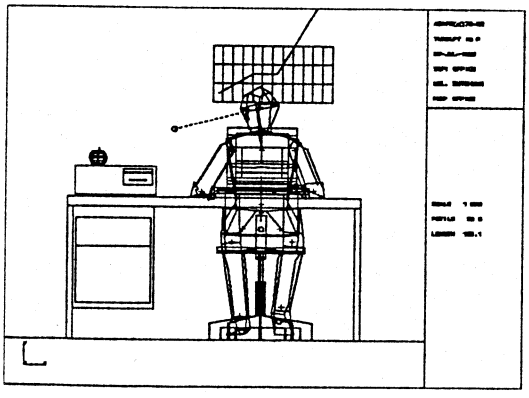


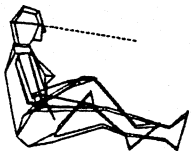
you first could tell the model to look at the apple while you are in side-view mode:



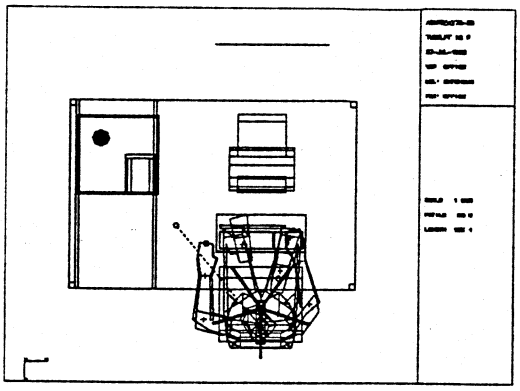
and then do the same while you are in front-view:



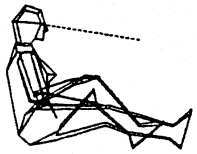




You can check with a top-view that the model's line-of-sight is really directed at the apple:



You will find that there is an even more easy way to direct the line-of-sight when menu item **VISION** will be discussed (page 1-69). You then will also see what the model "sees" !



C. Manipulation of the human model by entering values for the joint-angles of a body-member

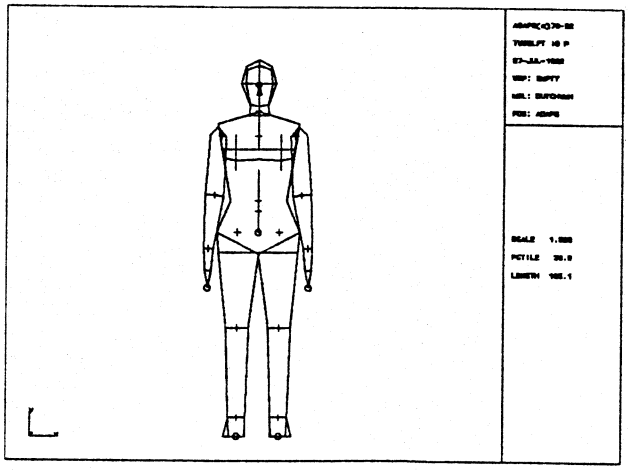
The ADAPS models contain twentyfive body-members or links (hand, lower arm, upper arm etc.; see chapter 3. The model).

Joint-angles define a link's oriëntation in a joint, relative to the connecting, more proximal link. If e.g. we want to oriënt the lower arm, we have to enter joint-angles relative to the upper arm.

All body-members or links, are accessed by activating the relevant marker: for most of them this is a plus (+) sign, for some (the ones that indicate an algorithm too) this is a tiny square (on the screen; in this manual something more like a circle). When you activate a marker it will change into an asterisk (*). You will then also see a sub-menu appear in the menu area, specific for that link. We already had mentioned this while describing the various algorithms. One last thing: body-members have their markers at their **distal end** ; the marker e.g. for the lower arm is situated at the wrist; the marker for the upper leg is situated at the knee etc.

Individual body-members/links are accessed by activating the markers at their **distal end.**

Let's have a look at this. Suppose you want to change the oriënta-tion of the model's right upper arm in the figure below:



Remember that the marker for the upper arm is located at it's distal end: the elbow. Activate this marker: place the cursor over the marker and click mouse. If activated, the marker will change into: * (if not, the cursor probably didn't come close enough; just try again). When activated you will see the following sub-menu appear:

```
=CHANGE POS.=
R UP. ARM
FI    =     0
THETA=    84
PSI   =     0
END
```

We will discuss what these angles stand for in chapter 3. At the end of the manual you will also find an overview of all the joint-angles.

Now click on over:

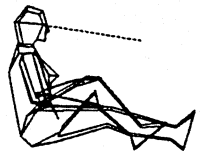
THETA= .

When done, you will notice that the sub-menu changes a little bit: the line with END now displays the excursion-range for theta of the upper arm:

```
=CHANGE POS.=
R UP. ARM
FI    =     0
THETA=    84
PSI   =     0
-44 <TH< 138
```

and you are prompted to enter a value for theta:

```
Input   THETA
>
```



Input **0** via the keyboard and give **<Enter>**.
You will see that the value is accepted (since inside the range) and echoed in the sub-menu:

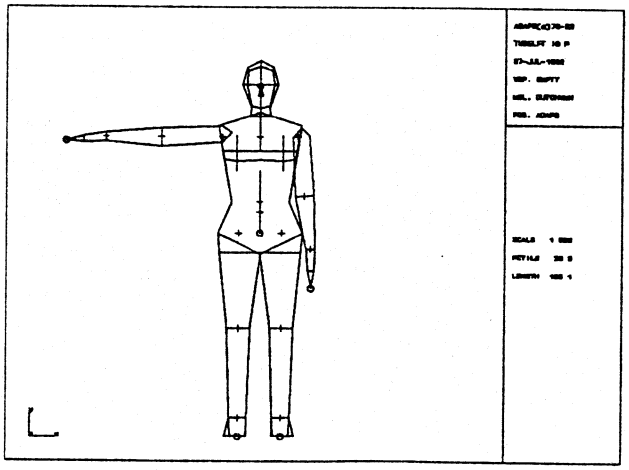
```
=CHANGE POS.=
R UP. ARM
FI    =     0
THETA=      0
PSI   =     0
END
```

The excursion-range for theta has disappeared and been replaced with END again. Sofar, the display hasn't changed: you might want to input values for the other angles as well.

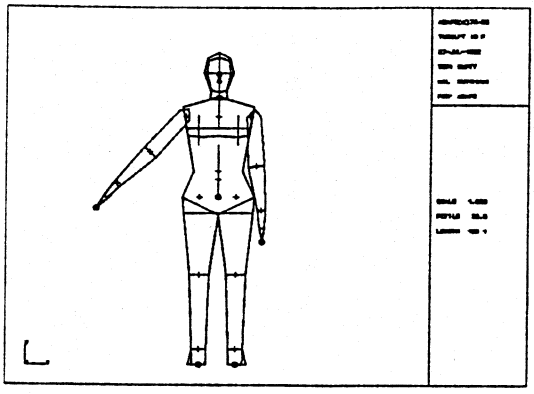
When you now click on over:

END

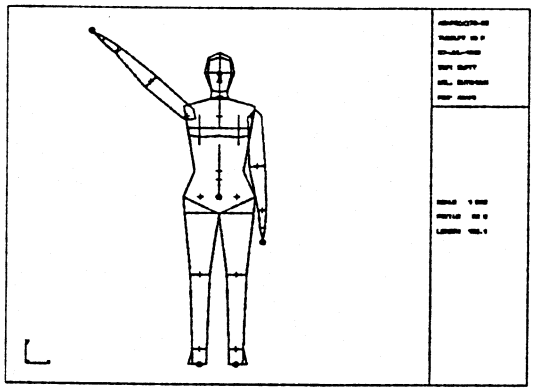
the model will be recalculated, using the latest values for the joint-angles. After the theta = 0 input the screen will look like:



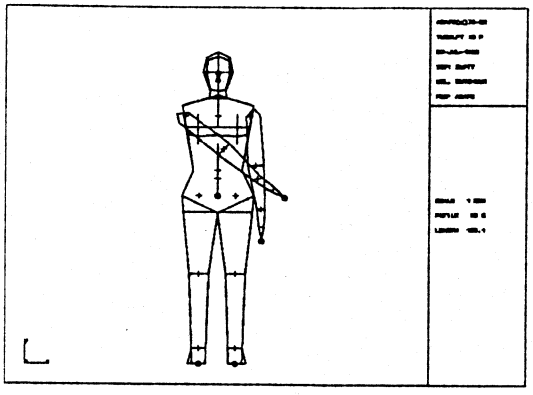
You can experiment a little: repeat and enter 45 degrees for theta:

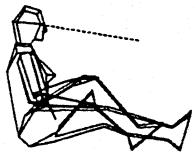


or enter one of the extremes (-44 degrees for theta):

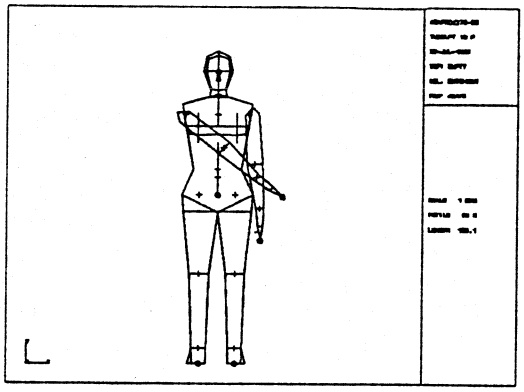


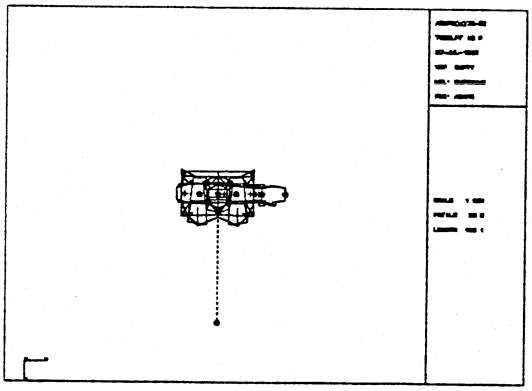
or the other extreme for theta (138 degrees):





If we combine this last picture with a top-view like below:

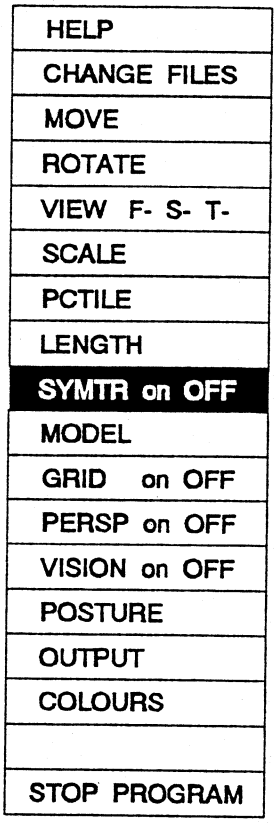




you will notice that this extreme for theta, although within its excursion-range, results in a quite impossible posture. The cause for this we discussed before: all possible values for the joint-angles (the excursion-range) in ADAPS are independent of values for the other body-parts: at the moment the model is not "aware" of its own body as a possible obstacle. Since this value (138 degrees) for theta is possible with the arm in front, as well as behind the torso, it can also result in the posture where the arm is sticking right through the body.

If you, while experimenting with the joint-angles, want to return to the **last-before-current posture** (you want to 'undo' a change in posture), there is a very easy way to accomplish this. We will come back to this later: menu-item **POSTURE** , **Restore** ; page 1-85.

At the end of this manual you will find an overview of all possible joint-angles



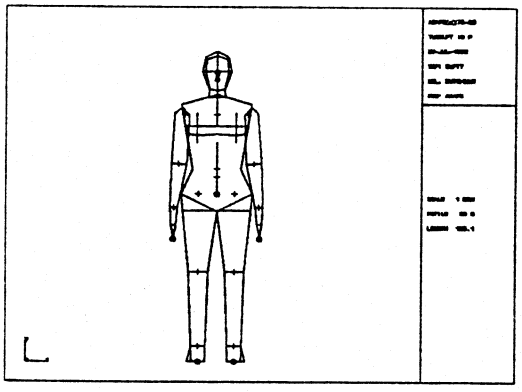
Menu item **SYMTR. on OFF** : with symmetry turned to ON changing the joint-angle of a body-member, will also change the same angle for the opposite (Left/Right) body-member.

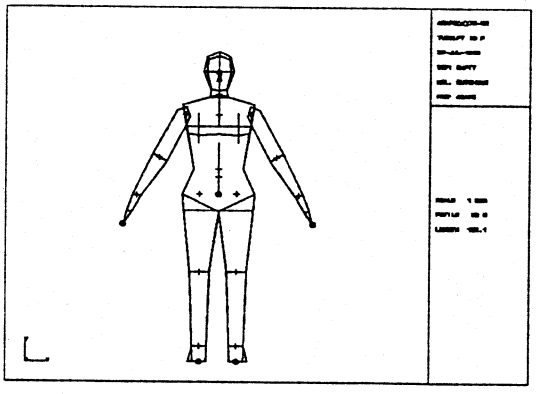
Let's return to the menu. When you have just started ADAPS you will see in the menu-area the default setting: **SYMTR. on OFF .**
Like GRID this option switches between ON and OFF. With symmetry turned to **ON**, changing the joint-angle of a body-member (link), will also change the same angle for the opposite body-member: what you input e.g. for the left upper arm, will be copied for the right upper arm.

Let's demonstrate this. Click on over **SYMTR.** ; (you will see the line change to: **SYMTR. ON off**). If we now input e.g. 60 degrees for theta of the right upper arm of the model in the picture below left:

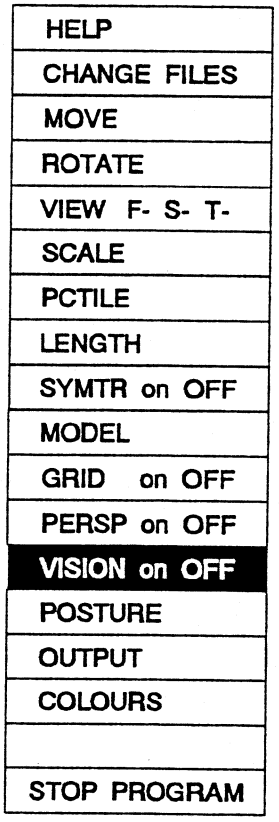
activate distal marker of the right upper arm
activate **THETA=**
input **60** via keyboard and give **<Enter>**
finish with **END**

you will then get the picture to the right:





Symmetry will be set to OFF , when using a reach-algorithm.

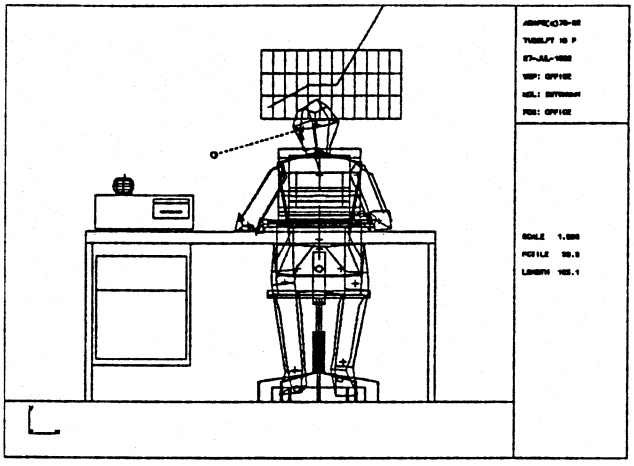


Menu item **VISION on OFF** : click VISION to ON to display the model's field-of-view;

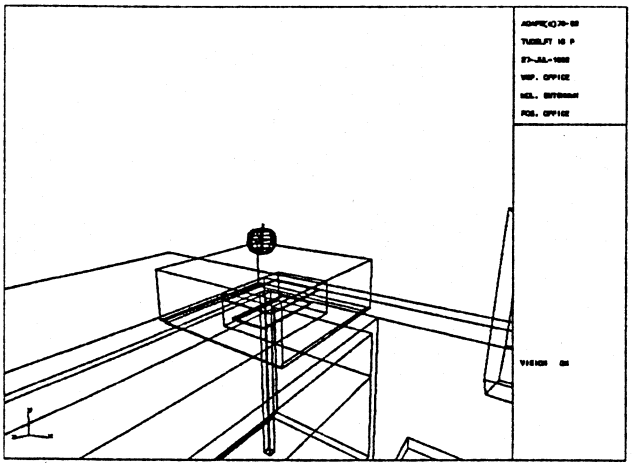
- direct the line-of-sight while staying inside this view-mode;
- reach with arms and legs when visible.

VISION

One of the nicest options in ADAPS is the next one: **VISION on OFF** (as you can guess this is the default setting). When you click VISION to ON , you see the model's field-of-view displayed. This, combined with directing the line-of-sight, enables you to inspect the workspace as if you were the model yourself. Remember we had let the model look at the apple in the office-scene a few pages back:



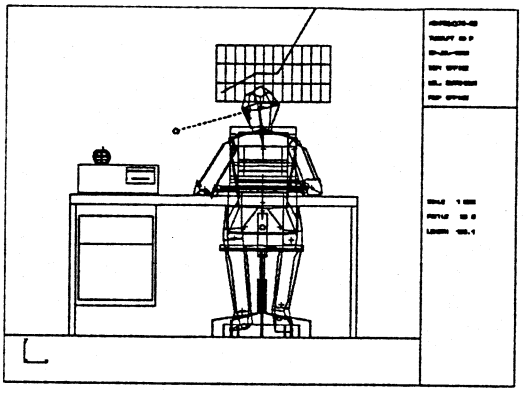
All you have to do now is click on **VISION** and you will see what the model sees:

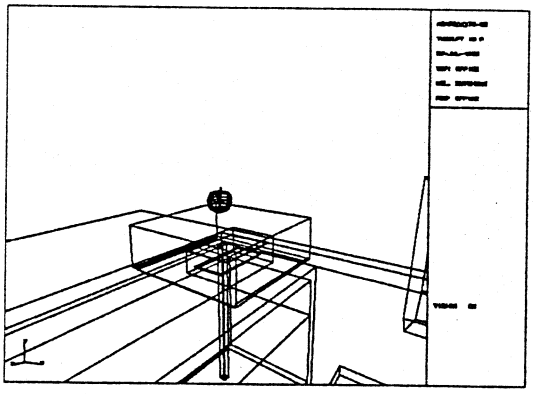


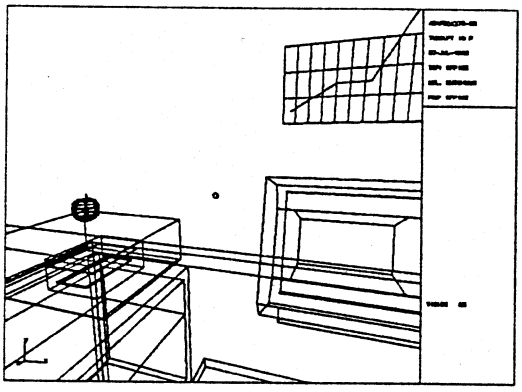
Another nice part of this option: the reach-algorithms and directing the line-of-sight, are still accessable **while staying inside** the **VISION ON** option.

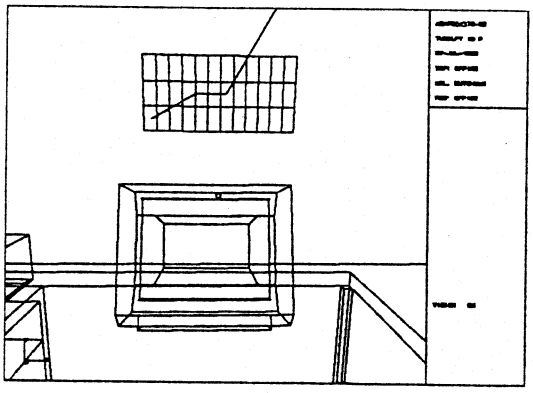
VISION

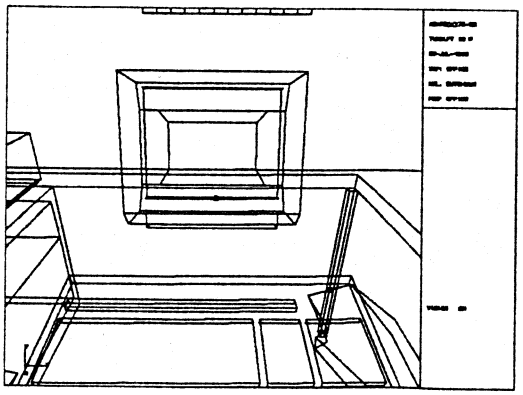
While you stay inside VISION, you **activate the marker** of the line-of-sight (it should be **exactly** at the center of the display area - in this case somewhere in the apple). You next get a sub-menu that we will skip for the time being, but in the display area you see the same message as before: **indicate new position or select menu.** So, just move the cursor a bit to the right and click on. Repeat this and check that you can get a series of views like:

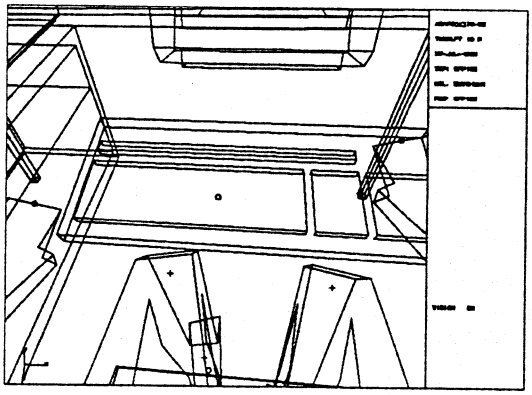






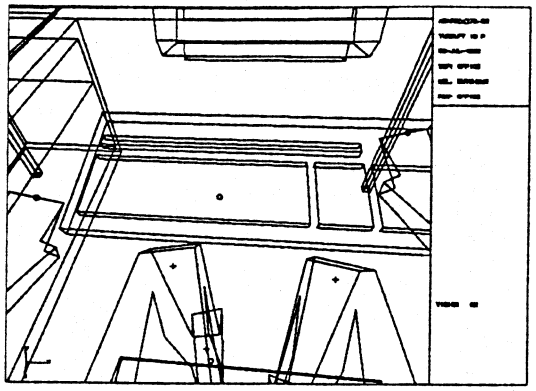




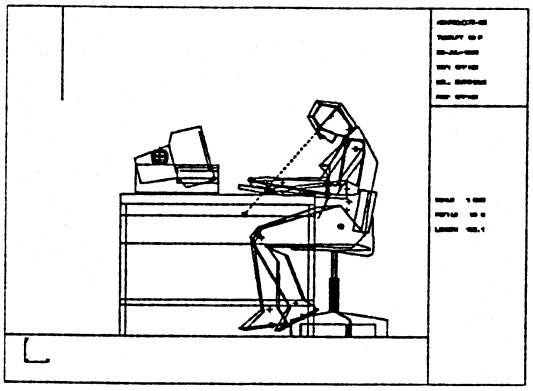


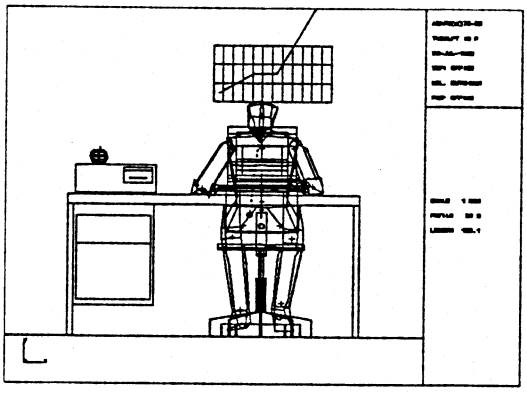
VISION

If you now click VISION to OFF again, you are back as observer of the general scene. However, while you made the model look around, its posture did change. This is what you last saw:



and this is now the model's new posture:



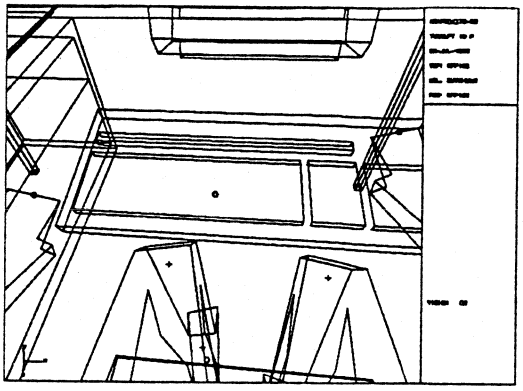


So, you could have reached this posture in two ways:

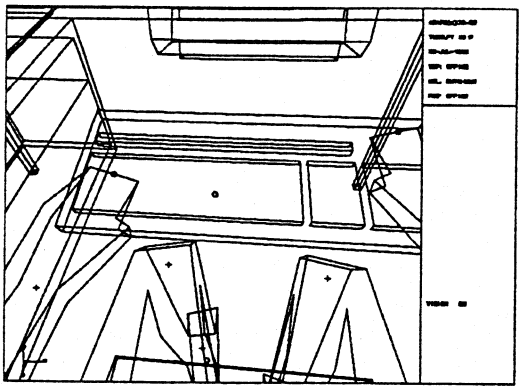
- VISION is OFF : by directing the line-of-sight, via e.g. a side-view and a front-view (as outside observer)
- VISION is ON : by looking around as if you were the model.

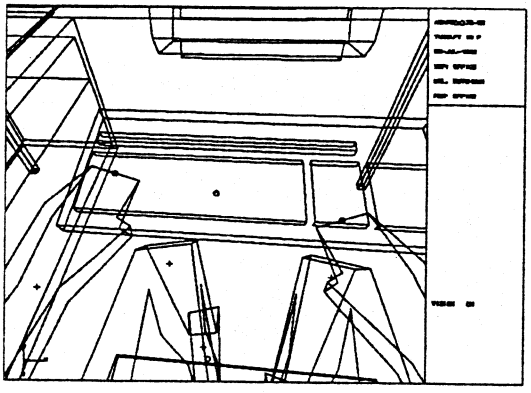
The reach-algorithms remain accessible with VISION ON : if you can see your (the model's) hands or feet you can position them by mouse.

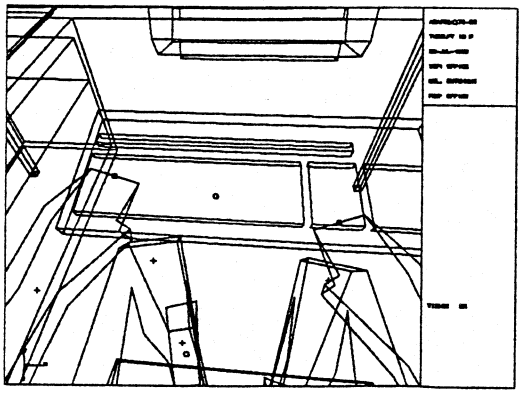
If you turn VISION to ON again:

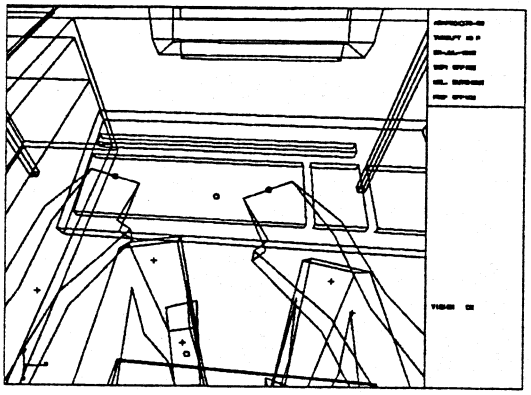


you can see the model's hands and feet. So, activate the end-markers and reposition them:



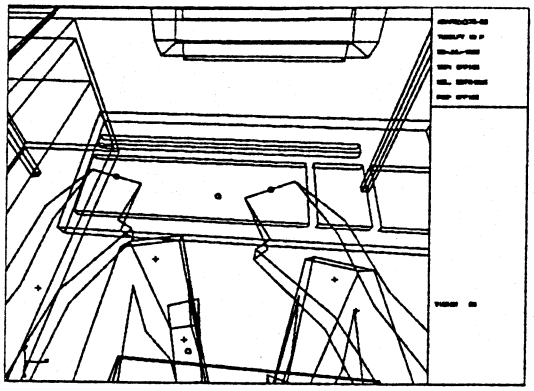


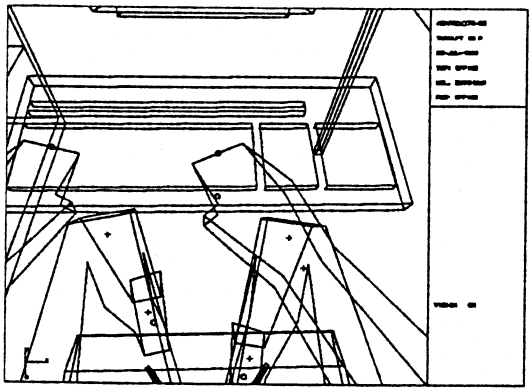


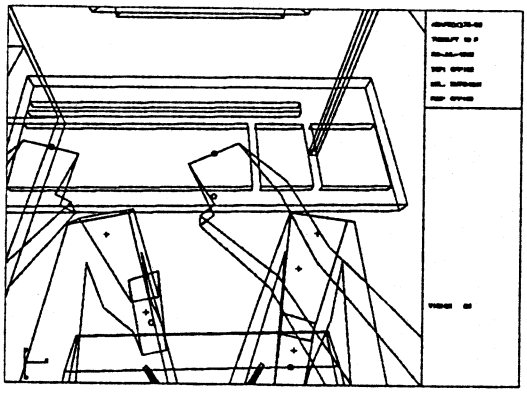


VISION

If you can't see an end-marker (as e.g. with the right foot in the last picture), activate the line-of-sight marker (that is always there, in the center of the field), and look in the proper direction (if possible) until the marker comes into sight and can be activated:



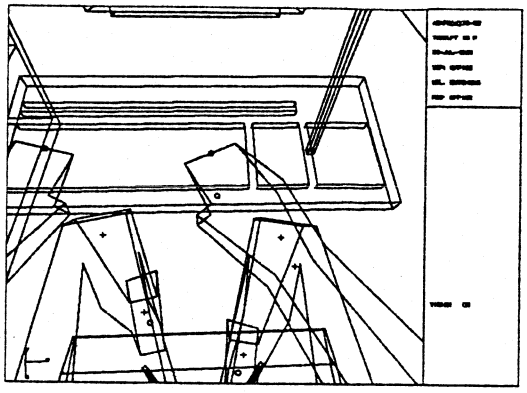


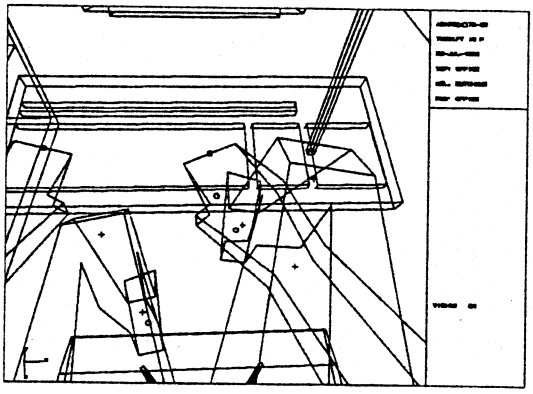


You should be aware however that moving the arms and legs while VISION is ON, takes place in a plane **perpendicular to the line-of-sight** (just as moving them when VISION is OFF takes place in the front-, side- or top-view plane). This might result in unexpected postures.

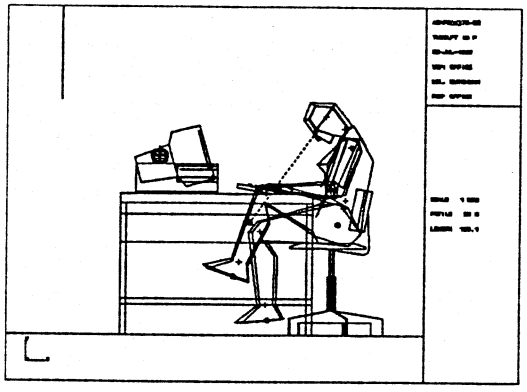
As algorithms get "smarter", we tend to project **all** of our human capacities onto the model: " Right, now place your foot over there. But **of course** keep it on the ground (the footrest etc.)". We certainly are not that far yet.

If for instance, we would like to stretch the right leg forward in the view below left we end up with something like in the picture on the right:

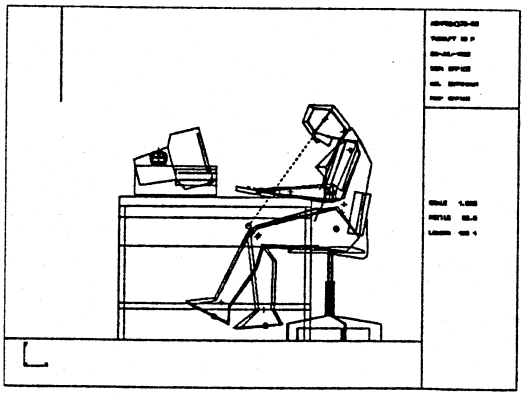




When we next turn VISION to OFF and choose a side-view we will get:



Although this can be corrected very easily (we activate the right foot again and shift it):

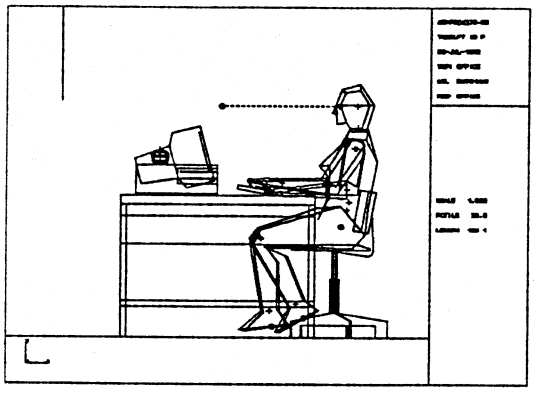


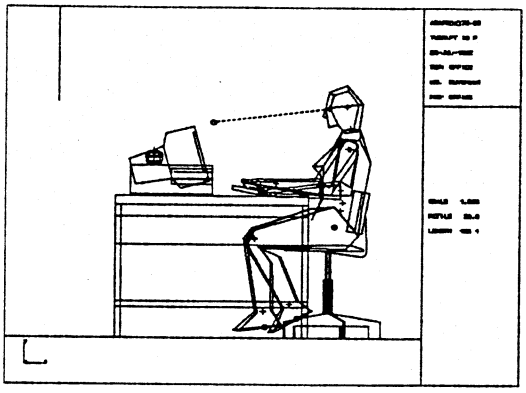
it wasn't what we expected to happen.

VISION

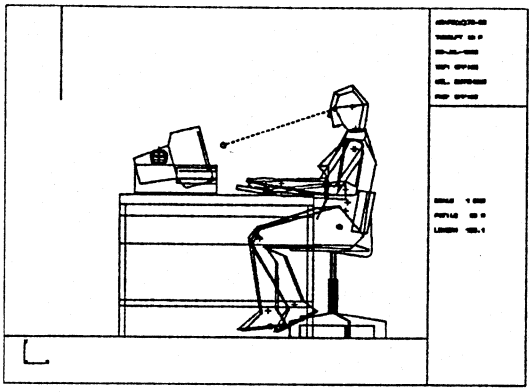
Remember that when you use the **line-of-sight** algorithm, the values for its joint-angles are **reset** to the default values (**looking straight ahead**), and only after that a new head/neck posture is calculated. This might result in a posture where the model bends its head too much compared with you, who would automatically also use the eyes. Since you can input values for the individual joint-angles in ADAPS, you can imitate this but it's good to know that you should do this **after** using the line-of-sight algorithm.

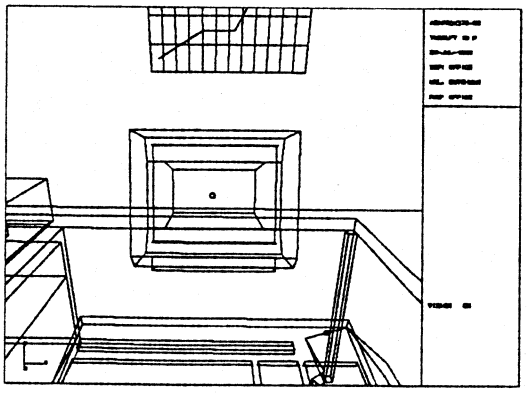
If we had e.g. the office scene like below left, we could first direct the line-of-sight to go just a little bit above the monitor:



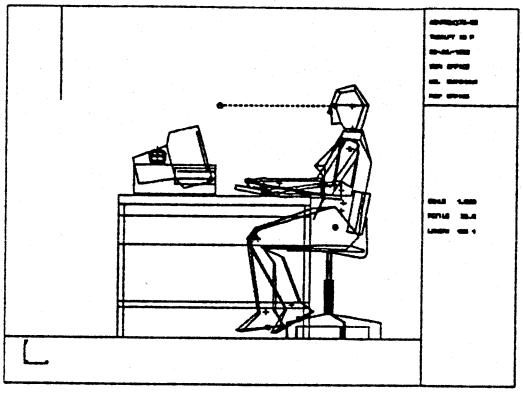


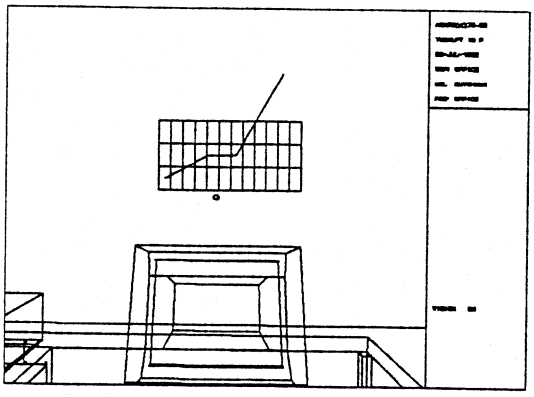
and then input values for the joint-angles: we activate again the end-marker of the line-of-sight, but next we go to the sub-menu and choose **THETA=** . We input something like **12** degrees for a downward view and give **<Enter>**. We finish with **END**. We then get the scene like below left and the corresponding field of view as in the right picture:



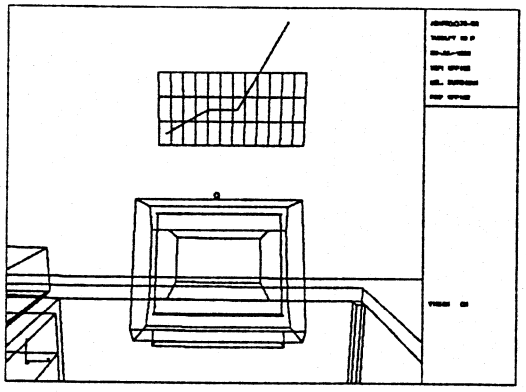


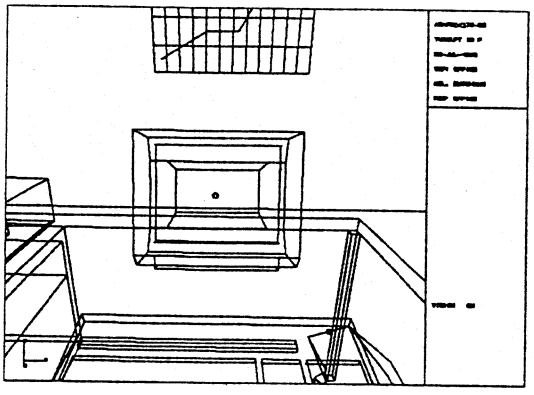
You get the same result of course, by first setting VISION to ON:





Then, while you stay inside this mode, direct the line-of-sight a little bit above the monitor (below - left picture); after this activate the line-of-sight again, but next choose **THETA=** in the sub-menu, input **12** degrees and give **<Enter>**. Finish with **END** and you will get the picture on the right:





In this way ADAPS can offer you some very handy tools for visual inspection of the workspace; however, it still needs you, a human user, to tell what's right and what is wrong in a specific posture.

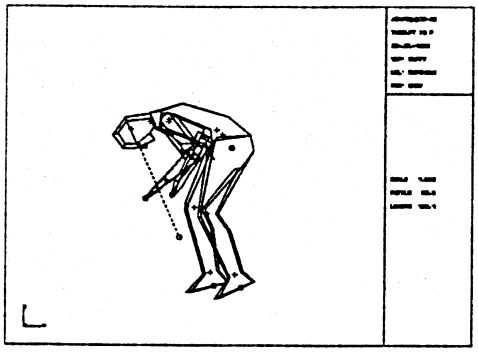
Some last words regarding vision: with VISION set to ON the ADAPS program displays the model's field-of-view. This is a pyramid with a 'vertical' top-angle of 60 degrees, with the top at the proximal end of the line-of-sight, centred along the line-of-sight.
The 'horizontal' top-angle is a little bit more than 60 degrees since the display-size is not a square. However, when you make a copy of the screen to a printer (see later under **MENU OUTPUT** , page 1-107), the print is set to the largest square that fits the printer; then both top-angles are 60 degrees.

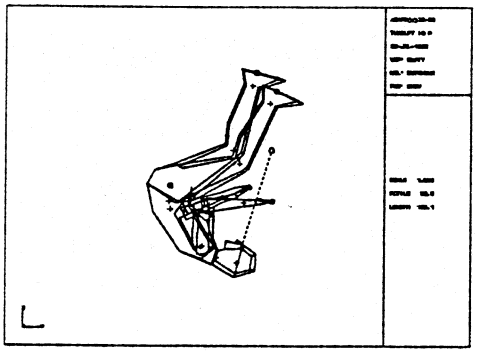
VISION

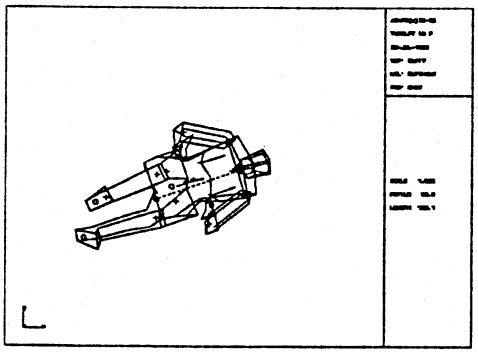
We have a nice question: what's **'UP'** with VISION ON ?

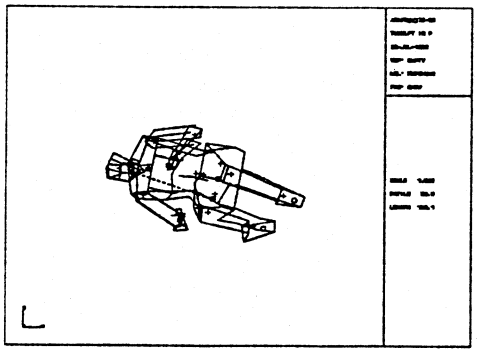
In ADAPS, 'up' means: from the eyes towards the eyebrows;
'down' means: from the eyes towards the nose.

This means that, given a certain posture of the model, **whatever the oriëntation** in the workspace:

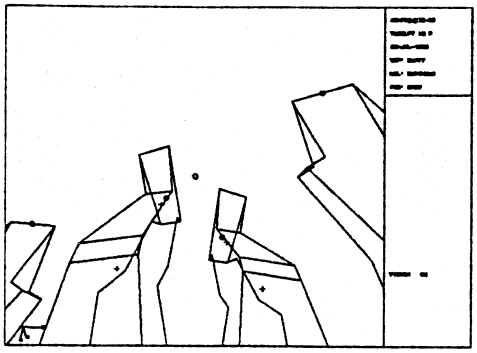


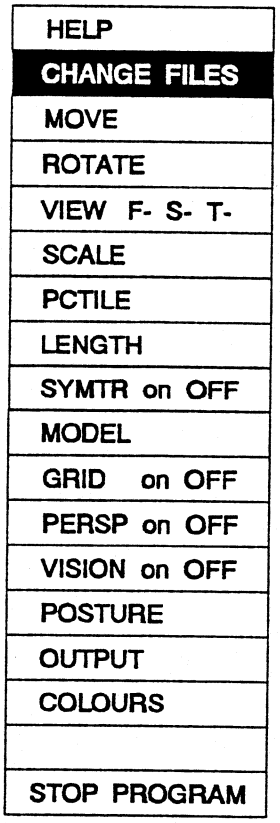






the model's field-of-view (of itself) will stay the same:





Menu item CHANGE FILES : **change** - workspace files
- model files
- posture files

create - new posture files

edit - workspace files

CHANGE FILES

At the start of ADAPS you were asked to choose three files:

- a workspace file,
- a posture file, and
- a model file.

With menu item CHANGE FILES you can do what the name implies: change models (to check your workspace-design for another population), change workspaces (to check design-alternitives) or change posture files (you would like to use a posture, stored in a different file than the one you started with). With menu-item CHANGE FILES you can also **create a new posture file .**

When you click on **CHANGE FILES** , you will get the next sub-menu:

=== FILES ===

WORKSPACE
MODEL
POSTURE
EDIT WSPFILE

Escape

Most of these choises are self-explanatory: choose WORKSPACE, MODEL or POSTURE, to get a **list of available files** in your choice and choose one of them. Workspace- or model files are then read and displayed, posture files are connected for reading or writing (storing) postures. When you choose EDIT WSPFILE you can edit the current workspace file (see also chapter 2. The workspace).
(Depending on your editor, you can now also enter DOS-commands: for instance to print file INFO.DMP while staying inside the program (see POSTURE, PRINT>FILE, page 1-94)).

When you choose **POSTURE** you can also **create a new posture file.** After this choice the sub-menu reads as:

KEYBOARD INPUT
UPDATE LIST
MORE (if you have more than 20 files)

Escape

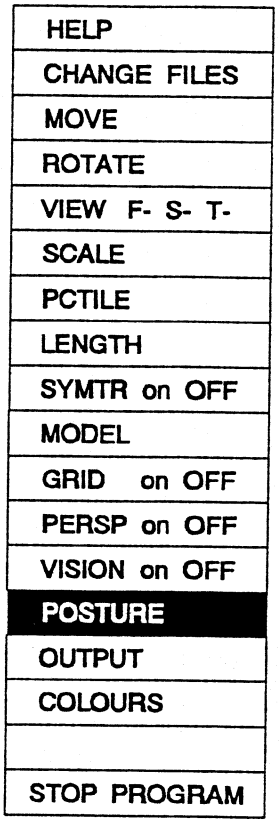
When you want to create a new posture file click-on **KEYBOARD INPUT.** You are then prompted for a filename:

Type filename (NO extension): .....

At this moment you can also input the name of an existing file, but when you input a **new name** (8 characters maximum) and give **<Enter>** a new posture file is available for you where you can store up to twenty postures. After this option, you are returned to the main menu and the screen you had before. You will see the name you gave to the new posture file displayed at the upper right, but the file doesn't, as yet, exist: it will be created as soon as you write (store) the first posture to it (see later under MENU **POSTURE, WRITE** , page 1-90).

Use **UPDATE LIST** in the sub-menu of **CHANGE FILES** to include the newly created files in the list of available files.

One last thing about changing models. Whenever you switch, their first posture is always set to default: standing up straight in front-view. Don't be afraid to 'lose' the last posture you had before you changed models: you can always restore that with menu **POSTURE, Restore.**



Menu item POSTURE : with **READ** you can retrieve a posture from the connected posture file

with **WRITE** you can save (store) the model's current posture in the connected posture file

PRINT > FILE gives you a readable file of all the joint's co-ordinates and all the joint-angles of the model's current posture. You can print that file (INFO.DMP) afterwards.

Restore acts as a kind of **'undo'** or 'OOPS' of the latest change in posture.

POSTURE

You really are getting somewhere now: you can display any scene you would like, in a number of ways; move the model around in the workspace, use the reach- and line-of-sight algorithms, input values for the individual joint-angles, in short manipulate the model into a functional posture for your specific workspace.

In ADAPS, 'posture' stands for two things only: the combination of all the joint-angles and the co-ordinates of the base-point ('**where** is the model'); posture does not mean the specific model itself or its percentile (see also POSTURE, WRITE , page 1-90).

The next step of course is to save the model's current posture, retrieve a saved posture, output posture-information or maybe undo a change in posture. The next menu item makes all this possible.

Click on **POSTURE.**
You will then get the following sub-menu:

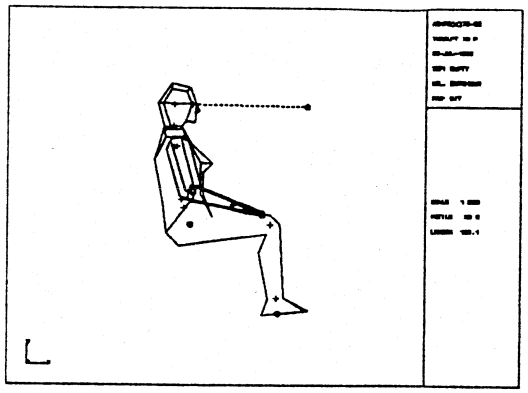
== POSTURE ==

READ
WRITE
PRINT > FILE
Restore

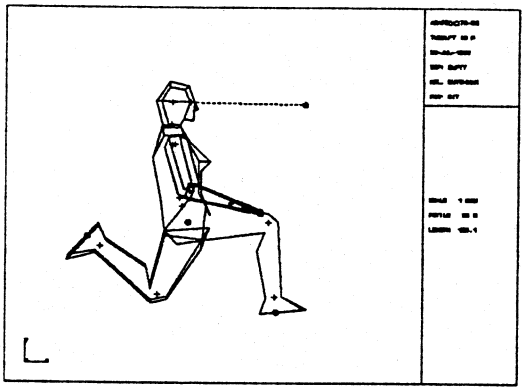
Escape

POSTURE
Restore

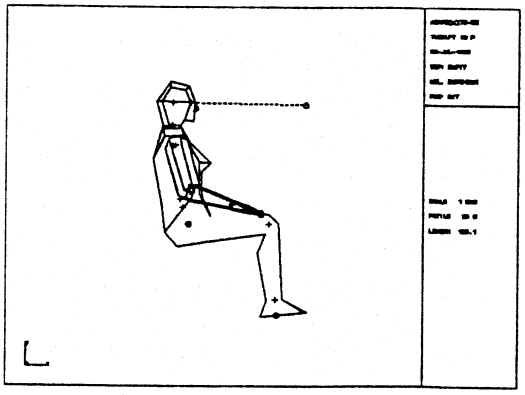
Let's start with Restore.
Suppose you had manipulated the model into the posture below:



and somehow activated a leg and positioned the mouse somewhere behind the model to get:

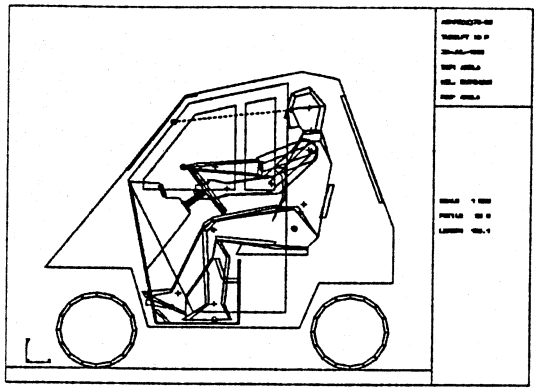


All you have to do now to get the first picture back is choose **POSTURE** and in the sub-menu choose **Restore.** The model will then be reset to the last-before-current posture:

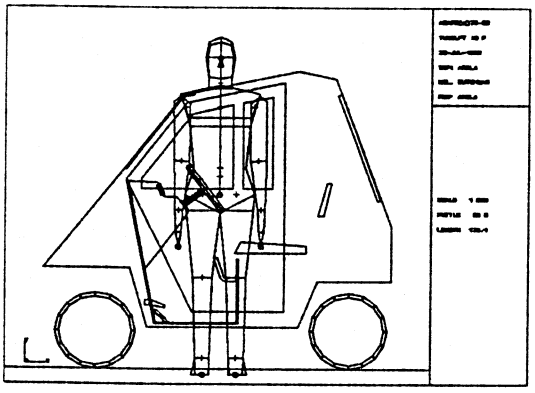


POSTURE
Restore

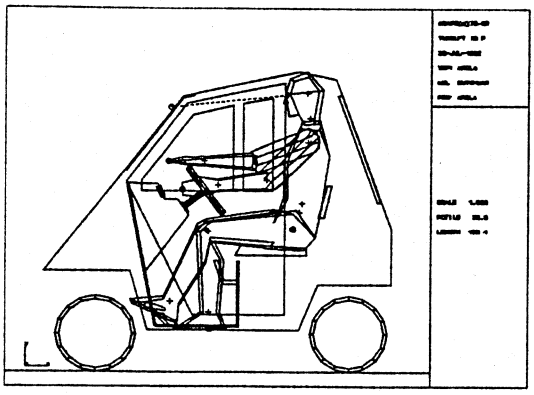
Another **use** of this option is **when** you have **changed models.**
Suppose you had manipulated a model into a functional posture for the next workspace:



and you would like to check this posture+workspace with the average dutch male model. Choose **CHANGE FILES**, choose **MODEL**, and finish with **DUTCHMAN.** You now get:



As you can see, the new human model's posture is set to default: standing up straight in front-view. All you have to do to get the last-before-current posture back is: choose **POSTURE** and **Restore :**



POSTURE
Restore

This example of an existing workspace (a small car for the physically impaired - designed for a *French* population), nicely demonstrates the problems another, taller population (like e.g. the Dutch) would encounter.

Remember that a posture consists not only of the combination of all joint-angles but also of the base-point's three co-ordinates. So if you have shifted the model, a **POSTURE, Restore,** will put it back in it's old place.

You cannot restore a posture while in the field-of-view mode (VISION is ON). Set VISION to OFF, to return to the normal view mode and then use the restore-option.

POSTURE
READ

As a first help we supplied you with a number of posture-files (ADAPS.POS , SIT.POS etc.), that were filled, each with twenty pre-set postures for sitting and standing.

When you started ADAPS, one of the files you could select was a posture file. That file is one of the files that is now connected (you can see their names at the upper right of the screen). From this connected posture file you can read (and write to) postures. We also have seen (MENU **CHANGE FILES**) how to connect other files.

Suppose at the start you had picked the posture file SIT.POS and now you would like to inspect and maybe choose one of its postures. All you have to do is click-on **POSTURE** and in the sub-menu click on **READ**. The screen then displays the list of twenty postures, each with a short (one-line) description, and it prompts you to make your choice.

It also offers you an Escape though, so you **dont't have to** make a choice: just click on **Escape** and you are returned to your old screen and posture.

If you **do** want to select a posture, click-on over the choosen line. You are then returned to your old scene with the main-menu but the human model is redrawn according to the new posture (remember that you can get the old posture back with: **POSTURE** and **Restore**).

So it's as easy as that:

click on **POSTURE**
click on **READ**
click on the **line** of the posture you want (or choose **Escape**)

and you are back with the chosen posture.

UNDO the new posture ?

click on **POSTURE**
click on **Restore**

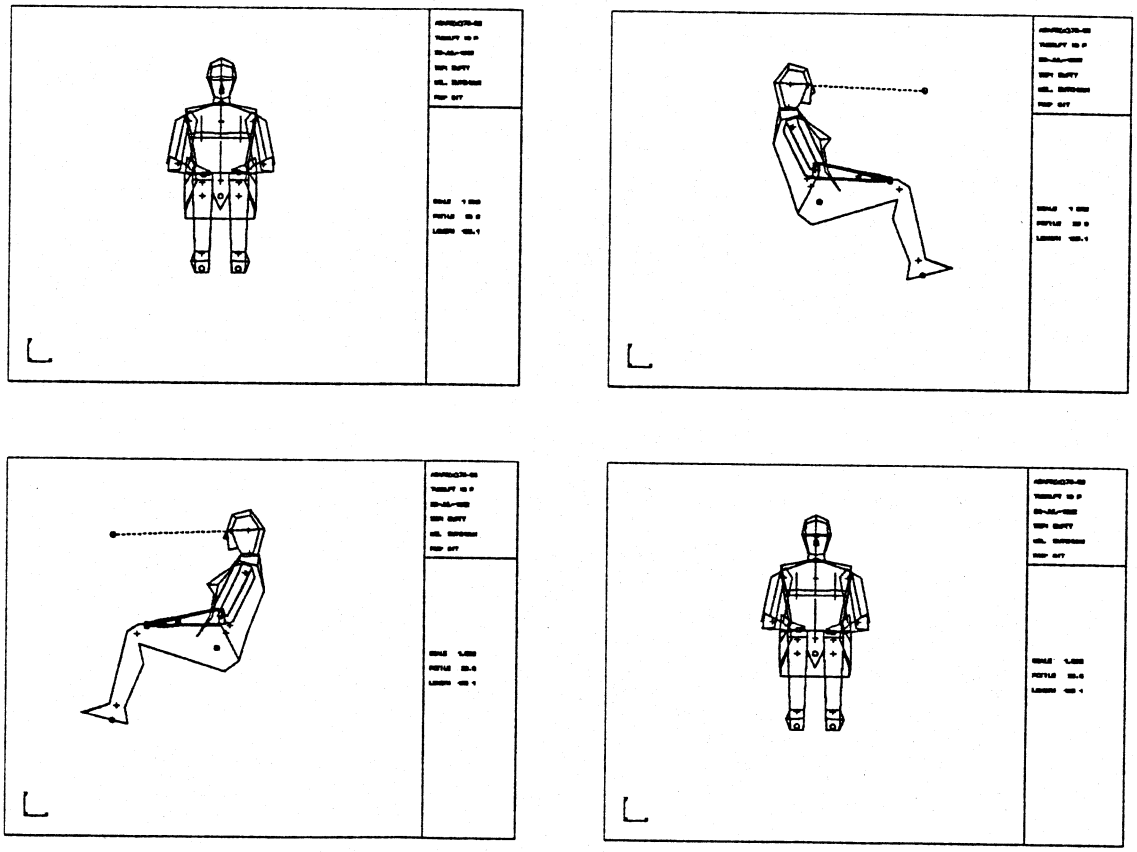
and you get the last-before-current posture back.

POSTURE
READ

Let's read a few postures from the posture file SIT.POS
After choosing POSTURE and READ, a list of the twenty postures in file SIT.POS are displayed. Among them you will find:

```
.....
5  as 1; back -30, front-view
6  as 5, from the right
7  as 5, from the left
8  as 5, from behind
.....
```

We can show these four postures here together (on your screen you would have to do it one at the time):



Please note that these are four **different** postures as seen from the same direction: a front-view of the scene. We will come back to this when discussing POSTURE WRITE (page 1-90).

POSTURE
WRITE

You have manipulated the model into a functional posture for the workspace that you are designing and you want to save your work.
With menu **POSTURE WRITE** you can save (store) the model's posture: all the **values for the joint-angles** and **the co-ordinates of the model's base-point.** With that data saved, it's very easy to reconstruct the model and in the right place. Something else is stored too: the way you last viewed it before storing: the **X-, Y- and Z-rotation** for the scene. We will come back to this later.

Nothing else is saved: neither the model itself, nor the length or the specific percentile that you used. What we do offer is the possibility to write a one-line comment that is 'tagged' to the posture you are saving. In this line you can put down any particulars that you think are relevant to recreate the scene.
Maybe something like:

p05, dutch female, reach. nose wheel contr., plane99, etc.

After reading this line and posture you then can load the appropriate model- and workspace files (if necessary), input the percentile and you are back where you wanted to be.
Postures are written (saved) to the connected posture file.
A posture file can contain twenty postures. If the file is full or you don't want to use it (remember that by choosing **POSTURE READ** (or **WRITE**) you can easily inspect it and return using **Escape**), you can connect another posture file or create a new file (use menu **CHANGE FILES**).

Let's demonstrate this. Suppose you have a functional posture that you would like to save in a **new** posture file. Click-on over **CHANGE FILES** . You then get the appropriate sub-menu we discussed before. Choose **POSTURE.** You will see the list of all available posture files, so you can check if the name that you had in mind for your new file already exists.
In the sub-menu choose **KEYBOARD INPUT.**
Next you are prompted to enter a name:

Type filename (NO extension): .....

Input your new name (e.g.) **qwerty** (max. 8 characters) and finish with **<Enter>**. You are returned to the main menu and the old scene but at the upper right of the screen you will see your new name echoed: POS: qwerty

---

Even though the name is there, the new file as yet doesn't exist: it will only be created as soon as you write the first posture to it.

So, choose **POSTURE** in the main menu and next choose **WRITE** in it's sub menu.

Your display now changes into:

```
  Select place to write posture or Escape

1 ...................not in use.........................
2 ...................not in use.........................
3 ...................not in use.........................
4 ...................not in use.........................
etc. up till line 20
```

Since this file is brand-new, all twenty possible places are empty (not in use) so you can place your cursor anywhere in this field and click mouse: the first free number (in this case of course no. 1) will be used to store your posture. After you have done this you are prompted at the bottom for a one-line description to go as a tag with the posture:

```
  Enter your description of posture:
.....................................................
```

If you have done that and finished with **<Enter>**, you are returned to your original screen. Meanwhile, the new posture file was created, your posture stored under the first free number and the text to go with it. You can check this very easily with **POSTURE**, **READ**, inspect what's there and use **Escape** to get back.

POSTURE
WRITE

Suppose 2: you want to 'borrow' a posture from the file SIT.POS and SIT.POS is not connected at this time (you have checked that its name is not at the upper right of the screen). The borrowed posture you would like to store in the newly created file qwerty.pos .

Here you go:

CHANGE FILES 'borrow' from SIT.POS
POSTURE
SIT.POS
The file SIT.POS is now connected so you can read the posture you want from that file:
POSTURE
READ
choose the posture you want

Your screen then will display the model in that posture.
You have the posture so you can store it now, but first you have to connect the file (qwerty.pos), that you want to store the posture:

CHANGE FILES connect qwerty.pos
POSTURE
choose your new file (qwerty.pos) if it's there ;
if not, choose **UPDATE LIST**
and then choose the file ;

qwerty.pos is now connected.

Store the posture:

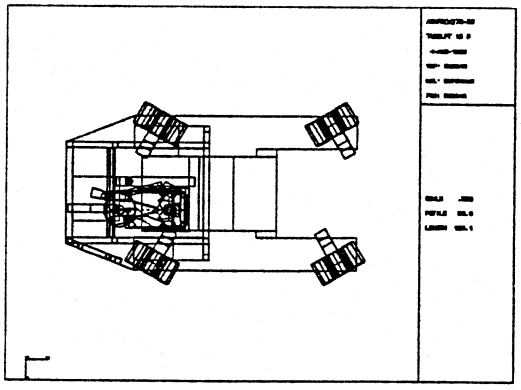
POSTURE
WRITE
choose the first free **line**
input your **description** and finish with
<Enter>.

Remember that you had a posture on your screen before the 'borrowing'. That posture disappeared (was changed for the borrowed one). You can easily get it back of course with:

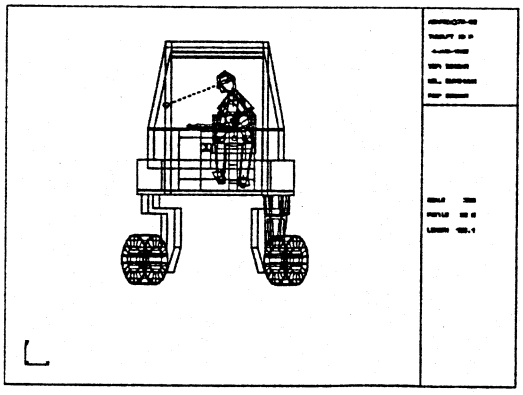
POSTURE,
Restore.

One last thing about saving (store or WRITE) a posture. Besides the co-ordinates of the model's base-point and the collection of all the joint-angles, the way you see the scene (the X-, Y- and Z-rotation) is saved. This means that when you save a special posture while you are (e.g.) in a side-view mode, those XR, YR and ZR values for a side-view (0,90,0) are saved too. It also means that when you next READ this saved posture, the model is reset according to that posture and the whole scene is reset to that side-view, even if you had a different view when you gave the READ command.

For instance, you could have saved a sitting posture looking forward in the forest truck while you were in a top-view mode:



You now change the posture, and you change the view to a side-view:

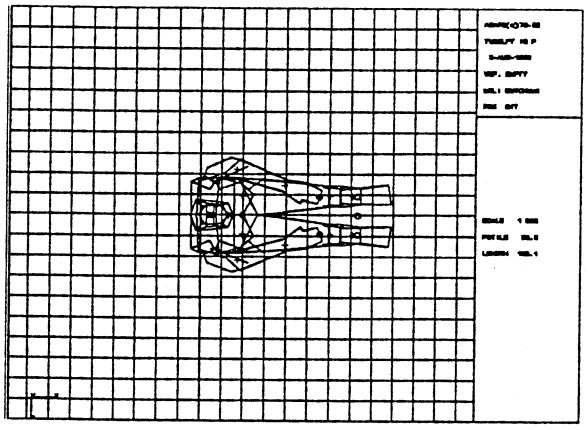


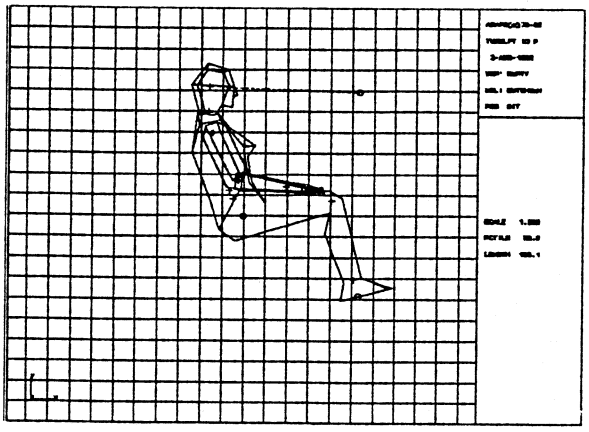
When you next give a POSTURE, READ of the first posture, you will get that sitting posture back and the scene will be reset to the top-view.

POSTURE
PRINT > FILE

The next item in the sub menu of POSTURE is PRINT > FILE .
You have a posture and you want a listing of the information that is contained: all the coördinates of the model's joints and all the joint-angles. When you choose this option the information is stored in the ASCII (readable) file INFO.DMP . You can print that file afterwards, or, depending on your editor, print it while you stay inside ADAPS (see EDIT WSPFILE in menu CHANGE FILES, page 1-80). INFO.DMP can contain information about **one posture** only.

As an example, suppose you had read from SIT.POS posture nr.6 (relaxed sitting, back 30 degrees backwards; in a side view from the right) and the p50 dutch female as human model. You then would have (in top- and front-view):





Now choose **POSTURE**, and **PRINT > FILE**. Print INFO.DMP (see above) and check that its contents reads as on the next page.

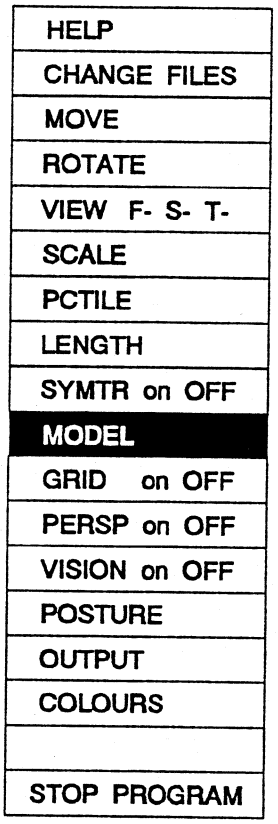
Contents of the file INFO.DMP :

ADAPS (c)79-92 TUDELFT IO 3-AUG-1992

MODEL: DUTCHWMN LENGTH = 165.10 PERCENTILE = 50.0

NR	NAME	LENGTH	XP	YP	ZP	FI	THETA	PSI
1	BASE-POINT	.0	-.4	.0	.0	-60.0	-90.0	90.0
2	PELVIC VERT	9.3	-5.0	8.1	.0	.0	.0	.0
3	LUMBAR	4.4	-6.9	12.0	.0	.0	5.0	.0
4	THORACIC	28.6	-14.3	39.7	.0	.0	10.0	.0
5	R SHOULDER	16.4	-14.3	39.7	16.4	-90.0	90.0	.0
6	R UP. ARM	26.2	-2.5	16.9	21.7	12.3	78.2	-28.7
7	R LWR. ARM	23.5	19.5	13.6	14.3	90.0	61.5	-70.0
8	R HAND	17.0	35.5	11.3	9.0	-90.0	.0	.0
9	L SHOULDER	16.4	-14.3	39.7	-16.4	90.0	90.0	.0
10	L UP. ARM	26.2	-2.8	16.8	-22.1	-11.7	77.5	28.4
11	L LWR. ARM	23.5	19.3	14.4	-14.5	-90.0	64.3	70.0
12	L HAND	17.0	35.3	12.6	-9.0	90.0	.0	.0
13	NECK HOR	1.2	-15.5	39.4	.0	.0	-90.0	.0
14	NECK VERT	10.9	-16.7	50.2	.0	.0	98.6	.0
15	HEAD	11.7	-16.2	61.9	.0	.0	8.5	.0
16	MID-EYE	9.3	-6.9	61.5	.0	.0	90.0	.0
17	SIGHT-AXIS	60.0	53.0	59.3	.0	.0	.0	.0
18	R PELVIC HOR	9.3	-.4	.0	9.3	-90.0	90.0	.0
19	R UP. LEG	41.5	40.5	7.2	9.3	70.0	90.0	.0
20	R LWR. LEG	39.3	50.7	-30.8	9.3	-90.0	85.0	.0
21	R FOOT	8.2	52.8	-38.7	9.3	.0	.0	.0
22	L PELVIC HOR	9.3	-.4	.0	-9.3	90.0	90.0	.0
23	L UP. LEG	41.5	40.5	7.2	-9.3	-70.0	90.0	.0
24	L LWR. LEG	39.3	50.7	-30.8	-9.3	90.0	85.0	.0
25	L FOOT	8.2	52.8	-38.7	-9.3	.0	.0	.0

This tells you a.o. the link's number, its name, its length in cm and the information concerning this particular posture.



Menu item MODEL : - display the human model (on/off)

- display a 2nd model as 'ghost'

MODEL

If you activate menu item **MODEL** , you will get the next sub-menu:

= MODEL =====
ON off

= 2nd MODEL =
STORE & DISPL
DISPLAY only

The first possibility is rather self explanatory: you can turn the model on and off - which only means that the model is displayed or not (e.g. in case you only would like to look at the workspace).

The second option however, implies that you can display a second model. This 2nd model can't be manipulated interactively like you can with the 1st, the current model. The 2nd model is something like a ghost, that you can make appear or disappear. It's just a way to display two models at the same time (e.g. what we did when we showed the p5 model together with the p50, when discussing menu PERCENTILE , page 1-39).

How can you make the 2nd model appear?

If you already have one (stored), that's very easy: just click on over **DISPLAY only** and it will be displayed (at least if it is inside your display area).
If you don't have one, or you would like to create a specific new 2nd model, you click on over **STORE & DISPL.**

What happens then is that a copy of the current model is made (and stored) and displayed with a different colour.

Since it's an exact copy, the 2nd model, the ghost, will overwrite the first model. The first is still there though! Just activate the base-point and shift the model. You then see two models: one that can be manipulated (you see the markers and the line-of-sight) and the 2nd model (the ghost), that's just there. If you now would click-on **MODEL** again, you would see that the sub-menu has changed just a little bit: there's a new line to go with the 2nd model item:

= 2nd MODEL =
STORE & DISPL
DISPLAY only
OFF

MODEL
2nd model

OFF tells you that you can turn the second model off, but without changing it: if you would click on over **OFF** , the 2nd model would indeed disappear (it would still be there but not displayed).

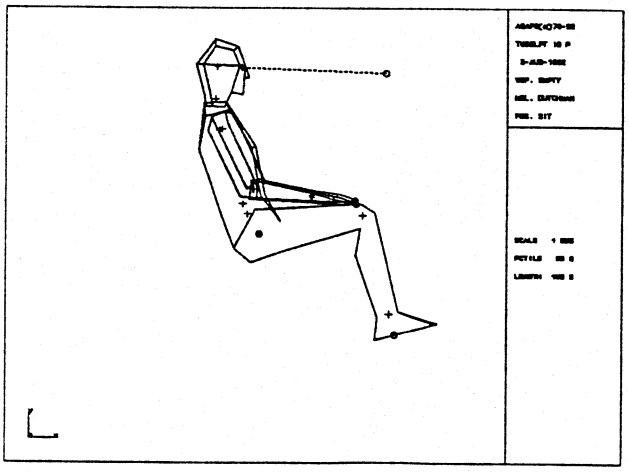
So: **STORE & DISPL** - creates a new ghost and displays it;
the old ghost is changed for the new one;
DISPLAY only - displays the old ghost.

As you can imagine, this option can be very illustrative when demonstrating population differences: for instance you can show a p95 male as 2nd model and then switch model files to combine it with a p5 female. Let's demonstrate this:

CHANGE FILES
MODEL
DUTCHMAN
PCTILE
95
<Enter>

We now have a p95 male from the dutch population. Its posture still is the default posture: standing up straight. We now get a sitting posture from posture file SIT.POS:

CHANGE FILES
POSTURE
SIT.POS
POSTURE
READ
line no.6 We then have:

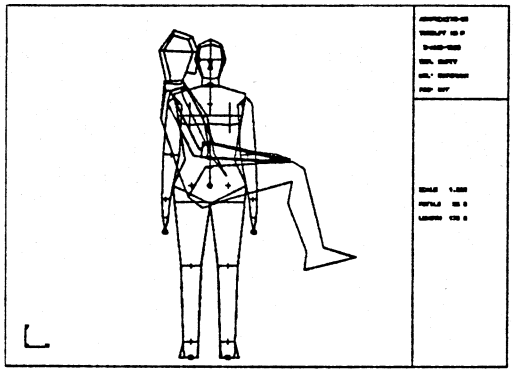


MODEL
2nd model

We store the model that we have, as ghost:

MODEL
STORE & DISPL

CHANGE FILES We switch model files
MODEL
DUTCHWOM Our display now looks like:

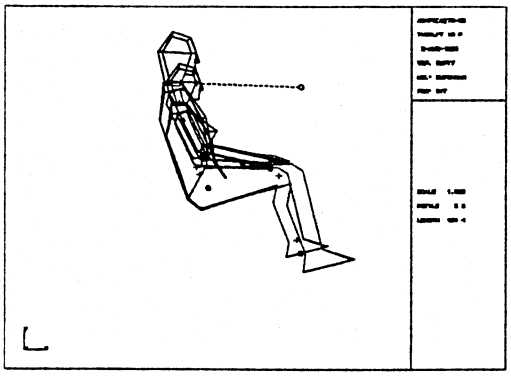


Change the percentile for the dutch female (at the moment it still is the p95 we had for the dutch male.):

PCTILE
5
<Enter>

We now would like to have her sit in the same sitting posture as the male. We could read that posture from SIT.POS (that posture file is connected), but it's even easier:

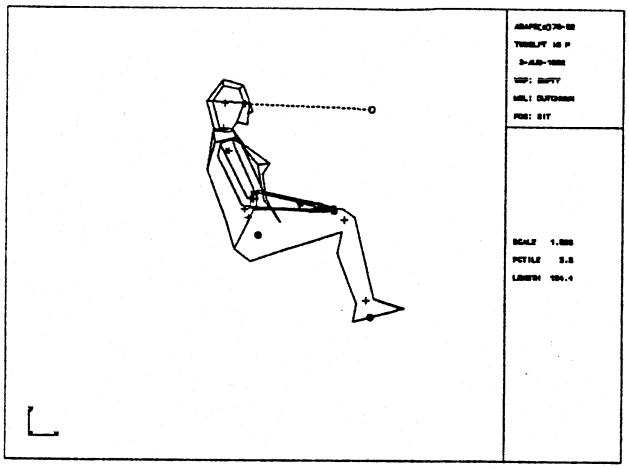
POSTURE
Restore which gives us the last-before-current posture:



MODEL
2nd model

You can check by the line-of-sight that the p5 female is the current model and the p95 male serves as the ghost.

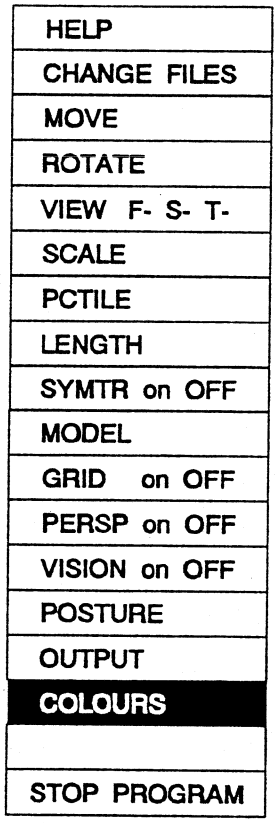
If you now would click-on again at **MODEL** and turned the 2nd MODEL to **OFF**, the male would disappear:



It would still be the ghost though, even if you quit the program. It will stay the ghost, right until you choose to change it with:

MODEL
  STORE & DISPL

The second model is stored in the binary file DUBMDL.BIN on your harddisk.



Menu item COLOURS : change default colour-setting for a number of display items

COLOURS

As you may have noticed on the screen, a number of display items like the human model, its markers, its line-of-sight etc. are displayed in colour. Colours for the various items have default settings but can be changed by you. If you select **COLOURS** the display screen will show you the list of items plus their default colour. The sub-menu area lists sixteen possible colours to use. Select the item whose colour you want to change by clicking mouse: an asterisk (*) will then be put in front of the item. If you change your mind, de-select the item by clicking mouse again; the asterisk then will disappear. After you have selected an item select your choice of colour. Your colour settings will be used until you stop the program.

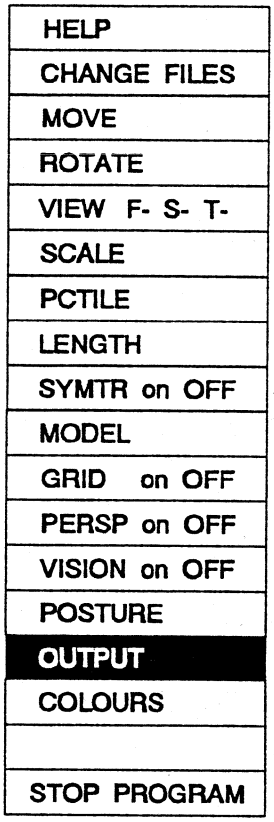
After you have chosen your colours, you can save your settings by selecting **SAVE** in the sub-menu (the settings are stored in a tiny ASCII (readable) file ADAPS.CLR). The next time you start the ADAPS program, these settings will be used. You can also recall your saved settings by choosing **RECALL**.

The default settings may be recalled by selecting **DEFAULTS**.

The sixteen possible colours are displayed in two series of eight: you can scroll through the colours by selecting **NEXT** or **PREVIOUS**.

Quit the **COLOURS** item by choosing **END** in the sub-menu.

Note: we found that some marker-types and colour settings will change when using ADAPS-output like metafiles (see later menu **OUTPUT** , page 1-107): the little squares that we use on the screen for displaying the joint-markers that can activate the special algorithms, will turn into circles when fetched e.g. inside Word Perfect (wp5.1 that we use for this manual). We also 'lose' the line-of-sight and the joint-markers, when using the default colour no.15: in wp this colour-number will be turned into the background colour. When preparing pictures as metafiles for this manual we changed the setting of the items with colour no.15 into no.16 .



Menu item OUTPUT : make a **hardcopy** of your display on the available hardware (printer, plotter), or
create **metafiles** (to be used in e.g. WordPerfect or AutoCAD)

OUTPUT

There are several possibilities to make a hardware copy of the display (depending of course on the available hardware).
In principle you can send your copy to a printer, plotter or metafile. A wide variety of printers and plotters is supported (e.g. it may be possible to create a (HPGL) plotfile which allows you to do your plotting after finishing an ADAPS session).

Metafiles can be processed by several DTP (Desk Top Publishing) systems:

WordPerfect 5.1 supports the metafiles ---.CGM
AutoCAD rel. 10 supports the metafiles ---.DXF

Contact your supplier for the right drivers for your hardware.

When you choose menu item **OUTPUT** you will get the next sub-menu:

== OUTPUT ==

PRINTER
PLOTTER
WP5.1 MFILE
ACAD MFILE

The pictures in this manual (using WP5.1) e.g., were made in the following way: we reset the items with colour no.15 to colour no.16 (see note with menu COLOURS , page 1-103).
We then choose:

OUTPUT
WP5.1 MFILE

We were prompted for a name:

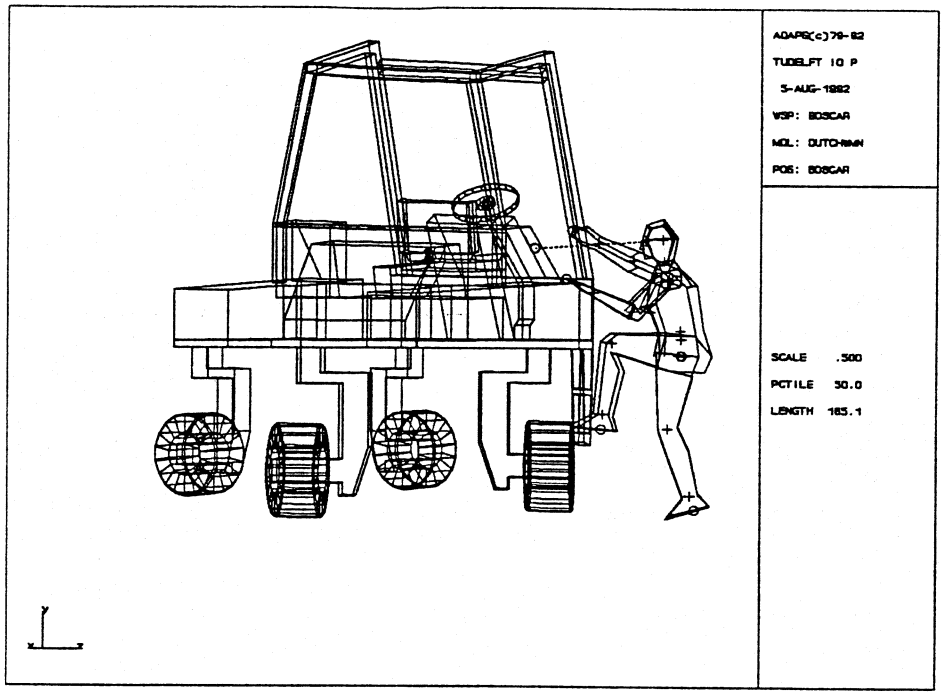
Enter name of metafile, without extension:

We now typed (e.g.): **truck** and gave **<Enter>** . The screen then showed:

Creating Computer Graphics Metafile truck.CGM , please wait...

After the metafile was created we were returned to the normal ADAPS screen.

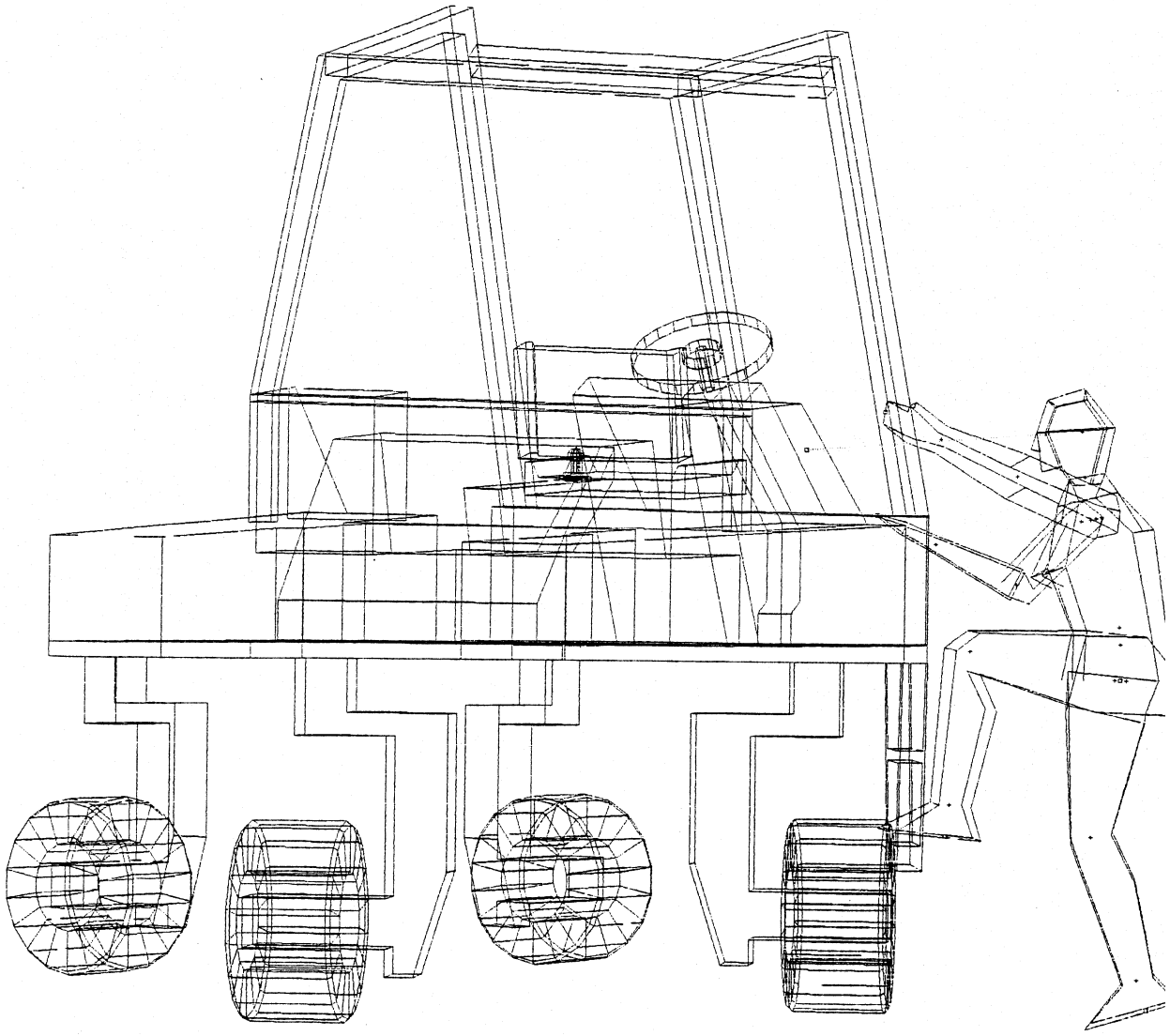
When typing this manual, we fetched the metafile truck.cgm (see a relevant WP-manual how to do this) at the appropriate place (here) to get:

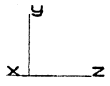


If we choose to output this same scene from the screen to our printer (a HP Deskjet 500) we select:

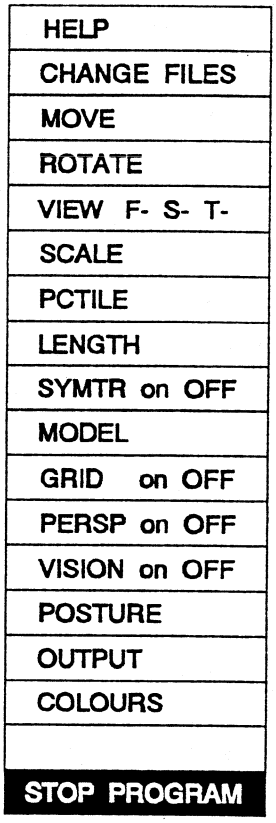
OUTPUT
  PRINTER

We then get a hardcopy as on the next page. Please note that the display area is now a square: the width is set to equal the display-height on your screen, so a little bit on the left and right side of your display will not be copied. Vertical and horizontal view-angles are now the same.





		BASE POINT		EYE POINT			
ADAPS(c)79-92	WSP: BOSCAR	X =	-131	X =	-124	SCALE	.500
TUDELFT IO P	MDL: DUTCHWMN	Y =	-4	Y =	59	PCTILE	50.0
6-JUL-1993	POS: BOSCAR	Z =	120	Z =	102	LENGTH	165.1



Menu item STOP PROGRAM : **Are you sure?**

YES - ends the program

NO - returns you to the main-menu

The last item lets you end an ADAPS session.

If you click on **STOP PROGRAM**, a sub-menu shows:

Are you sure?
YES
NO

This gives you the possibility to reflect and/or correct:

do I really want to stop ? (it might be that STOP PROGRAM was activated by accident).

did I save the last posture ?
(the **last posture** of the model is **NOT** saved automatically - you have to do it yourself).

If you click on **NO** you return to the program and can continue as you like.

When you click on **Yes**, the program really stops.

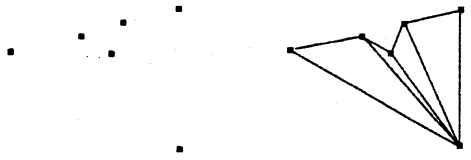
2. The Workspace

In order to visualize a workspace-design, ADAPS needs the file that contains the description of the workspace: the workspace file.

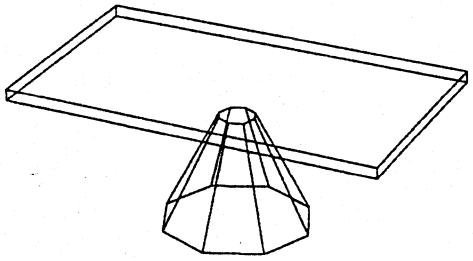
A **workspace file** contains the (coded) description of a workspace. It should be an ASCII (readable) file; its name with the correct extension: <name>.**WSP**

This description can be done in two ways, each of which with its own advantages:

- **linestream** (**ls**) - you define 3D-points in space and the lines that should connect them:



- **assembly** (**as**) - you assemble - pre-defined - graphical elements (blocks, cones etc.) into bigger entities that are called **objects** ; objects can then be controlled (shifted and/or rotated) as seperate units :



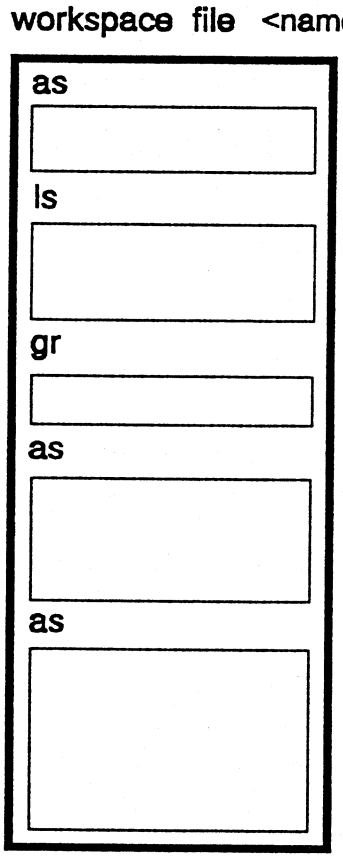
Next to these two description modes, you can also define a

- **grid** (**gr**) - you incorporate a grid of your own liking as a visual help in the workspace.

A linestream definition allows you more freedom, while an assembly is more efficient to code. Grids can be very usefull as fixed references for measurement purposes.

All three can be combined in a workspace file (see the figure below), but not mixed: you cannot treat (e.g.) a grid as part of an object in an assembly.

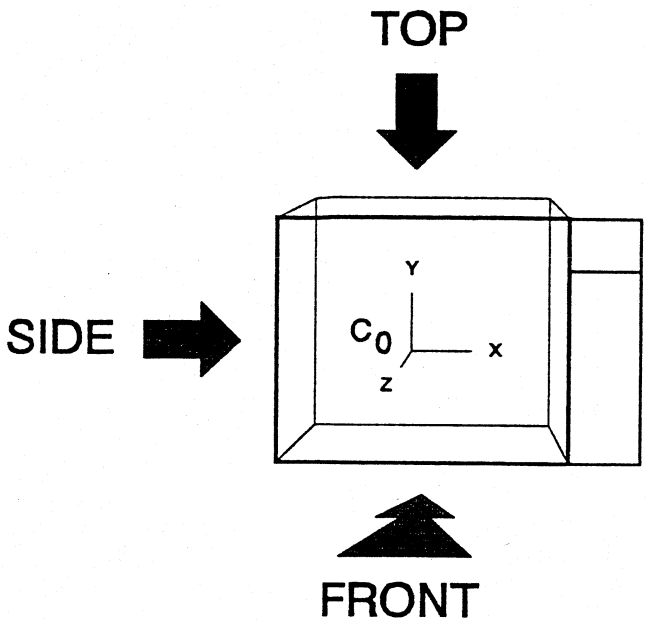
A workspace file could thus start with (e.g) an assembly, followed by a linestream, a grid and end with two more assemblies:



Switching the order of these combinations in the workspace file doesn't matter; it only influences the order of drawing: what is drawn first on the screen and what next.

Inside an object of an assembly though, we have a strict hierarchy. We shall come back to this later (page 2-17).

Before we start with the seperate description codes, we will show the display space again:



This basic co-ordinate system C_0 of the display space, with its origin at the center of the display area, is also used to describe the workspace:

X-axis to the right;
Y-axis upwards;
Z-axis towards us.

This is what we called the FRONT-view.

Note: you can of course define your workspace-design in such a way, that in this FRONT-view, you will see the **side** of your design.

2.1 **Linestream definition**

The definition of a linestream description is quite straightforward:

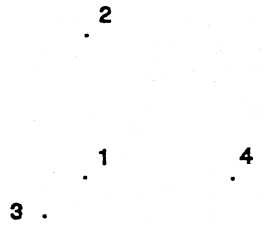
- you define 3D-points in local space (C_i);
- you define the lines that should connect them.

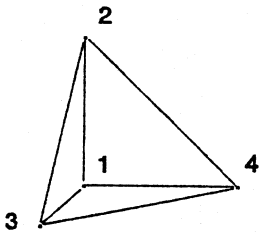
Note: the use of C_i enables you to shift and/or rotate a linestream part of the workspace relative to C_0 ; we will come back to this later (page 2-6).

Lines are straight lines between the 3D-points. A new line always continues where the last one ended. This means that to continue the linestream at a new starting point, without showing a line to it, we must draw an **invisible** line to that new point. You can code for this with a negative sign in front of the new starting point's number. For example:

2	means:	from where you are, draw a line to point no.2
-2	means:	from where you are, draw an invisible line to point no.2 ('jump' to point no.2)

If you had e.g. the four points as below left and you wanted lines between them as below right:





the next linestream might serve:

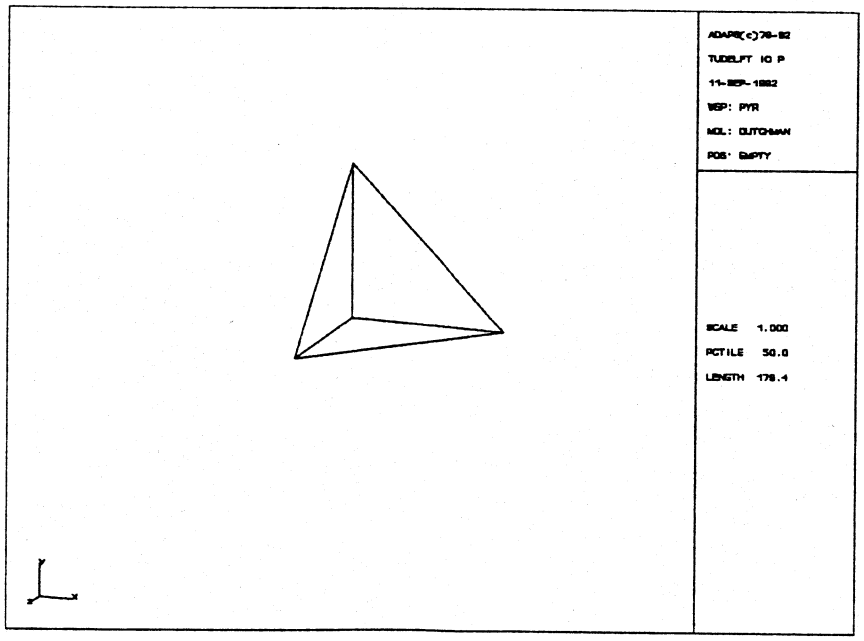
```
-2
3
1
2
4
1
-4
3
```

Note: other combinations of course, might do just as well.

Let's have a complete example.
Codes are on the left, comments on the right:

```
ls                          code for linestream;  no leading blanks
0  0  0   0  0  0           no shift or rotation of C_i relative to C_0
4                           number of 3D-points
0   0   0                   X  Y  Z  co-ordinates of point  #1; in cm
0  50   0                                                   #2
0   0  50                                                   #3
50  0   0                                                   #4
-2                          start of ls:   jump to point  #2
3                                 draw a line to point  #3
1                                   ..          ..      #1
2                                        etc.
4                              == one point per line ==
1
-4                                       jump to point  #4
3
9999                        endcode  for linestream
```

If we would enter this in a normal ASCII (readable) file and give that file a name and proper extension: **<name>.WSP** and chose this file when inside ADAPS, we would then get (with appropriate view settings), a view like:



We now come back to the remark we made earlier about defining the 3D-points in local space: co-ordinate system C_i .
Remember the second line in the linestream definition ?

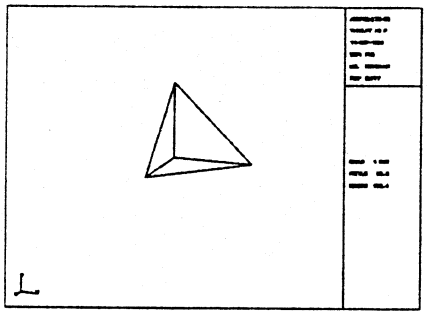
```
                    ls
These six values:   0  0  0     0  0  0
                    |  |  |     |  |  |
                    X- Y- Z- shift  and  X- Y- Z- rotate
```

allow you to shift and rotate your local space C_i relative to the basic display space C_0 ; if all values are zero, C_i equals C_0 .
This implies that we can define our linestream parts of the workspace in the most convenient way, and after that shift and rotate them (in that order) to a final end-position and orientation.
For example:

```
ls
0  0  0   0  0  0
```

as before

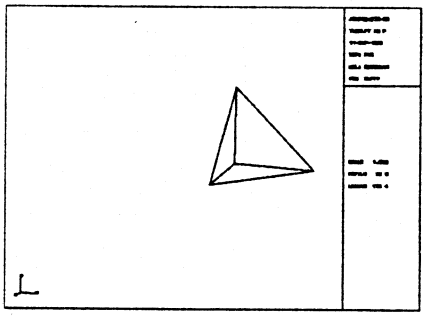
will give:



```
ls
40  0  0   0  0  0
```

as before

X-shift of C_i will give:

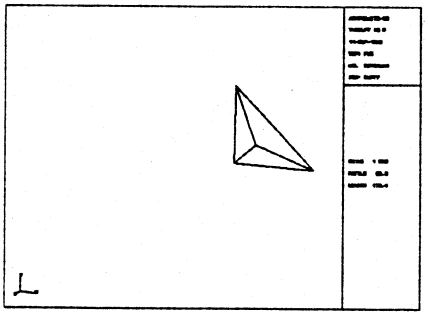


and

```
ls
40  0  0   0  90  0
```

as before

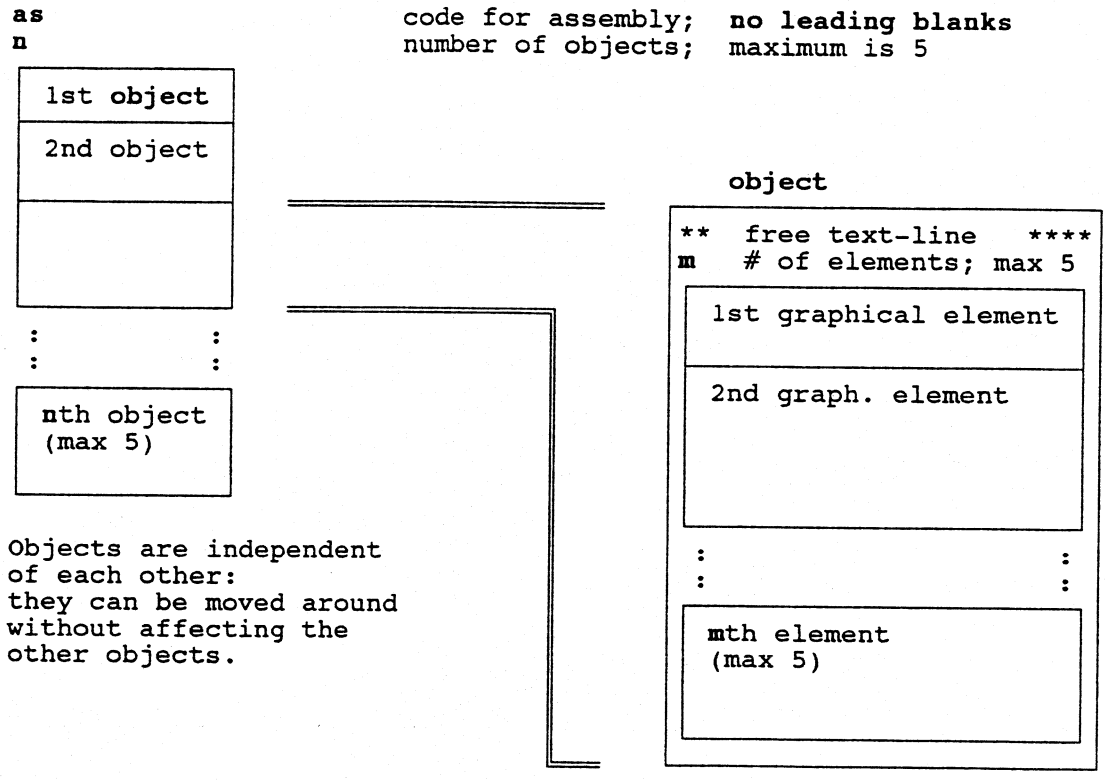
X-shift and then Y-rotate of C_i will give:



2.2 assembly definition

An assembly definition of (parts of) a workspace, allows you to assemble pre-defined graphical elements (blocks, cones, extrusions etc.) into bigger entities: **objects.** These objects can then be controlled (shifted and/or rotated), as seperate units.

The elementary form of an assembly definition of a workspace looks like:



Objects are independent of each other: they can be moved around without affecting the other objects.

An object consists of an **assembly of pre-defined graphical elements.**

Inside an object exists a strict hierarchy.

We saw that an object consists of a number (max. 5) of pre-defined graphical elements in a hierarchical order. This means that shifting and rotating an element only influences the place and orientation of elements lower down in the hierarchy.

We could e.g. define a table as consisting of a table-top and four legs. If we would specify the top first and the legs further on, the table-top would have the highest ranking in hierarchy. To move the complete table around, we would then only have to change the appropriate values for the table-top and the rest of the table would follow:

object

table-top
1st leg
2nd leg
3rd leg
4th leg

move table-top: moves the whole table

In the same way:

object

table-top
1st leg
2nd leg
3rd leg
4th leg

move 3rd leg: only moves 4th leg too

In ADAPS you can use four types of pre-defined graphical elements:

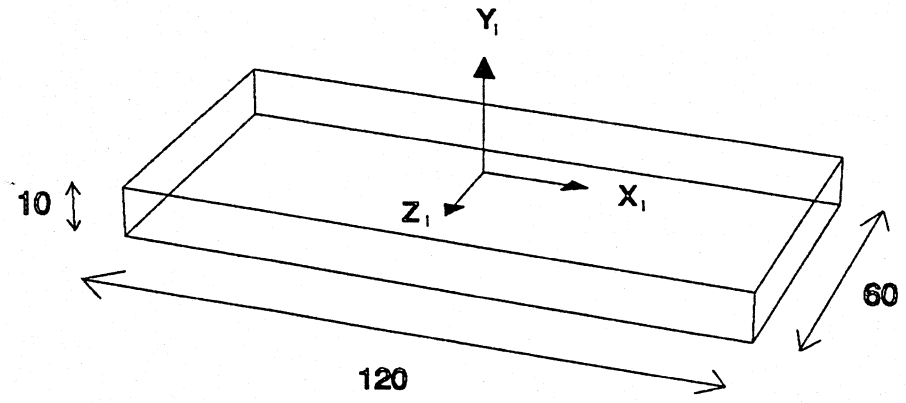
block	
cone	(which can become a cylinder)
rotation -sweep	(rotate around X-, Y- or Z-axis)
extrusion-sweep	(extrude along X-, Y- or Z-axis)

block

Local co-ordinate system C_i is at **the center of the top-plane.**

Example:

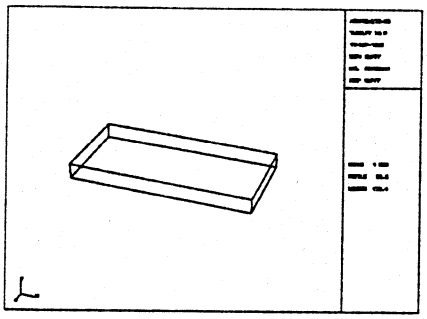
bl	code for block; no leading blanks please
0 0 0 0 0 0	no shift/rotation of C_i relative to C_0
120 10 60	X-, Y- and Z- dimensions of block (unit is centimeter)



You can shift and rotate (in this order) C_i relative to C_0 :

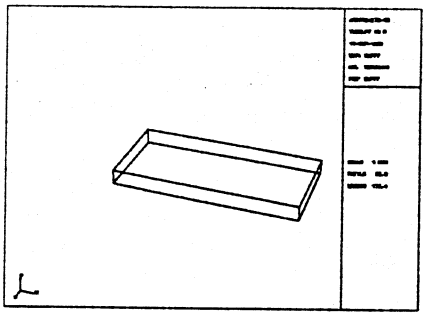
```
bl
0  0  0   0  0  0
120  10  60
```

will give:



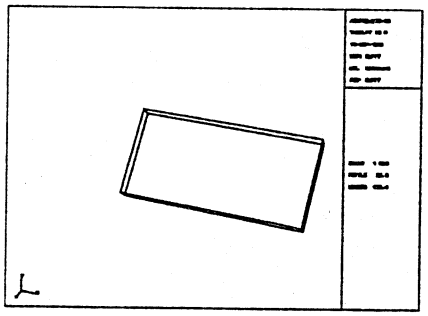
```
bl
30  0  0   0  0  0
120  10  60
```

will give:



```
bl
30  0  0   45  0  0
120  10  60
```

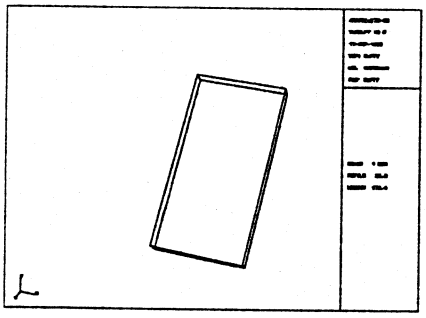
will give:



and

```
bl
30  0  0   45  90  0
120  10  60
```

will give:



Check that rotations are always executed in the same, fixed order:

first the X-rotation is applied;
then the Y-rotation around the **new** Y-axis;
and only then the Z-rotation around the **last** Z-axis.

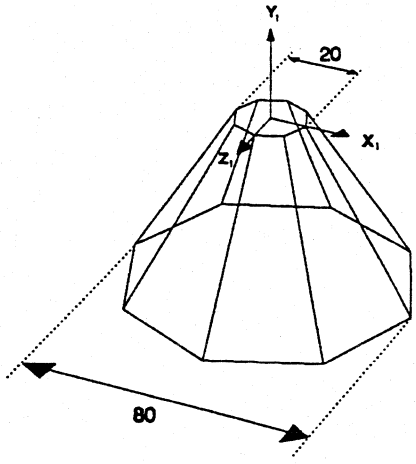
cone

Local co-ordinate system C_i at the centre of the top-plane.

Example:

```
ke                                  code for cone; no leading blanks
0  0  0   0  0  0                   no shift/rotation of Ci relative to Co
8  20  80  50
```

- number of sides n (3 ≤ n ≤ 512)
- diameter circumference top-plane; in cm
- diameter circ. bottom-plane; cm
- height ; cm



Check that by making both diameters equal, you get a cylinder. If you also set the number of sides to four, you will get something like an upright square beam.

Shifting and rotating C_i relative to C_o works in the same way as with a block.

rotation-sweep

A rotation-sweep is drawn by rotating meridian-points around a chosen axis (X-, Y- or Z-axis) from a start-angle until an end-angle.

The local co-ordinate system C_i can be chosen freely: you define C_i **implicitly** via the co-ordinates of the meridian-points.

Example:

```
rx     (ry or rz)
0  0  0    0  0  0
4  10    -30    270
```

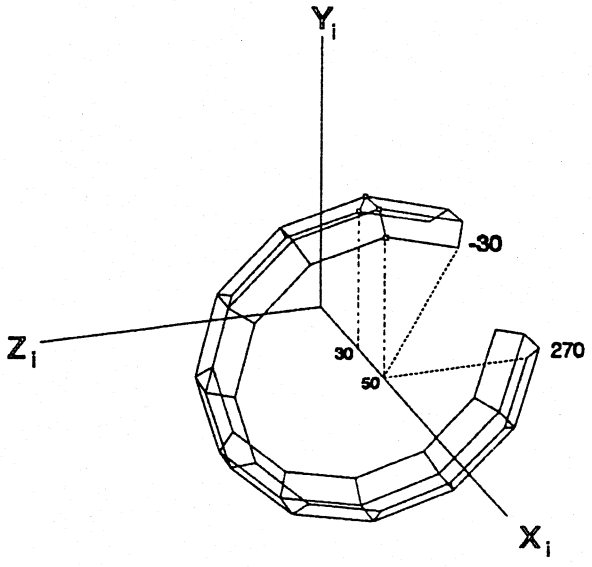
code for sweep around X- (Y- or Z-) axis
no shift or rotation of C_i relative to C_0

- no. of meridian-points mp (2≤ mp ≤64)
- no. of sides ns (3≤ ns ≤64)
- start-angle of sweep h1
- end- angle of sweep h2 (h2 > h1)

```
30  40
35  46
45  46
50  40
```

mp lines with meridian-point co-ordinates (in centimeter):

X	Y	for sweep around	X-axis
X	Y	..	Y-
X	Z	..	Z-



Check that we would get the same illustration with:

```
rx
0  0  0   0  0  0
4  10  -30  270
30  40
35  46
45  46
50  40
```

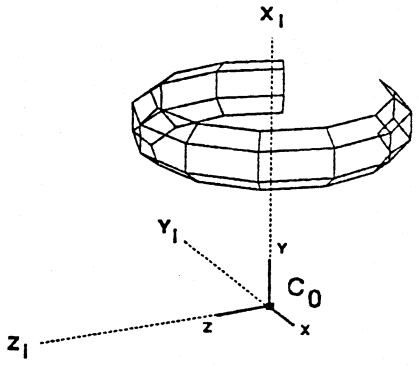
or

```
rx
40  0  0   0  0  0
4  10  -30  270
-10  40
-5  46
 5  46
 10  40
```

There is a difference of course: by entering **different** co-ordinates for the meridian-points, we define a **different** local co-ordinate system C_i . This difference becomes apparent when we not only shift, but also rotate C_i relative to C_0 :

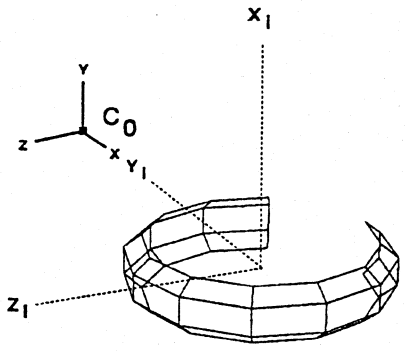
```
rx
0  0  0   0  0  90
4  10  -30  270
30  40
35  46
45  46
50  40
```

will result in:




```
rx
40  0  0   0  0  90
4  10  -30  270
-10  40
-5  46
 5  46
 10  40
```

will result in:



This freedom of choice for our local co-ordinate system C_i can be used to define parts of our design in such a way, that they rotate along their natural axes (e.g. wheels, pedals, levers, seat back-rests etc.).

extrusion-sweep

An extrusion-sweep is drawn by extruding a 2D-profile in the chosen direction (along the X-, Y- or Z-axis).

The local co-ordinate system C_i can be chosen freely:
it is defined implicitly via the co-ordinates of the profile points.

Example:

```
tz    (tx or ty)
0   0   0    0   0   0
6   70  -30
```

code for sweep along **Z-** (X- or Y-) axis
no shift/rotation of C_i relative to C_o

number of profile points np (3≤np≤64)

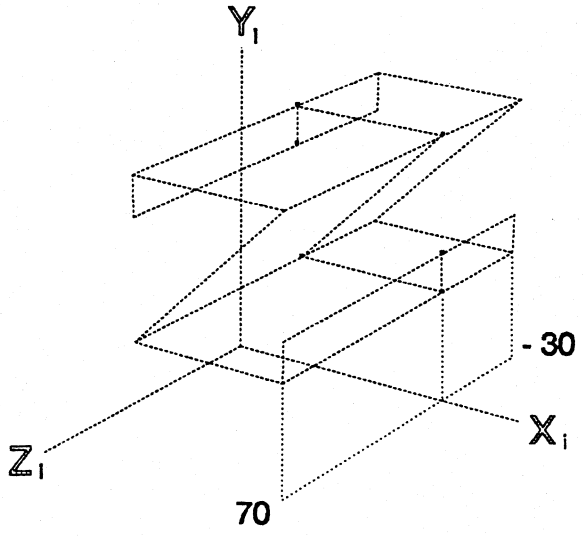
Z- (X-/ Y-) co-ord. of front-plane (cm)

Z- (X-/ Y-) co-ord. of back-plane (cm)

```
20   60
20   70
60   70
20   30
60   30
60   40
```

np lines with profile point co-ordinates (in cm):

Y - Z	co-ordinates for	X-sweep
X - Z	..	Y-
X - Y	..	Z-

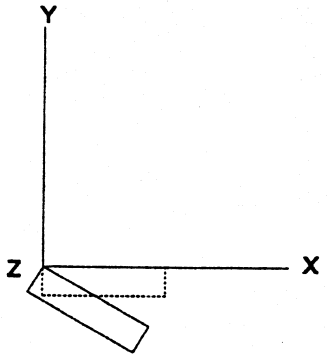


As with the rotation-sweep, the local co-ordinate system C_i is defined implicitly via the co-ordinates of the constituant points. This can be used to define your design in a more natural way.

Remember the first graphical element we discussed, a block ? That has a **fixed** local co-ordinate system C_i at the center of its top-plane and so would locally rotate at that center. By using an extrusion-sweep definition however, we can get the same graphics as for a block but now with C_i where we want it:

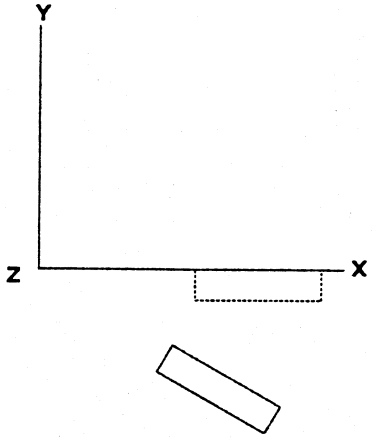
```
tz
0  0  0   0  0  -30
5  20  -20
0     0
40    0
40  -10
0   -10
0     0
```

will give:




```
tz
0  0  0   0  0  -30
5  20  -20
50    0
90    0
90  -10
50  -10
50    0
```

will give:



Both cases could be defined with a block-representation too; however, we would need to calculate by hand the exact shift needed to get the block's C_i in the right place.

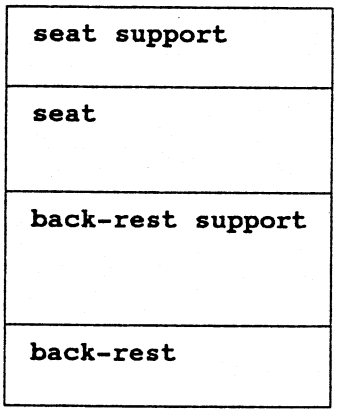
Using the hierarchy inside an object

We have seen that an assembly definition of the workspace implies using objects, that can be manipulated (shifted and/or rotated) as seperate entities.

Inside an object however, there exists a strict hierarchy: what is defined first has a higher ranking than what comes next. We can use this hierarchy when we define objects in such a way that their constituant parts behave as natural or functional as possible.

We could for instance define an object "chair" like this:

**** chair example **
4



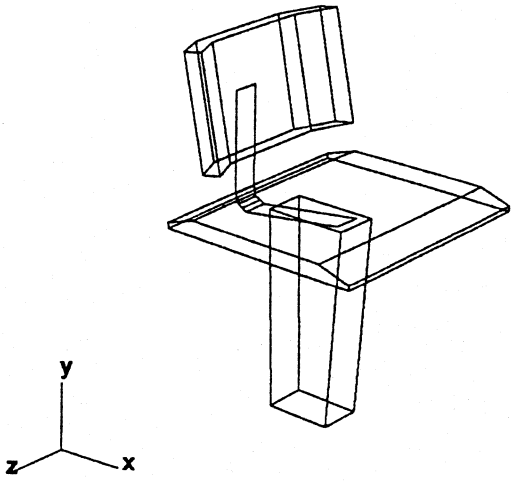
Because of the hierarchy, we can now move the chair around in the workspace by only shifting the seat support: the rest of the chair would follow.

In the same way a shift and/or rotation of the back-rest support would then only influence the back-rest.

In the following pages we will give some examples.

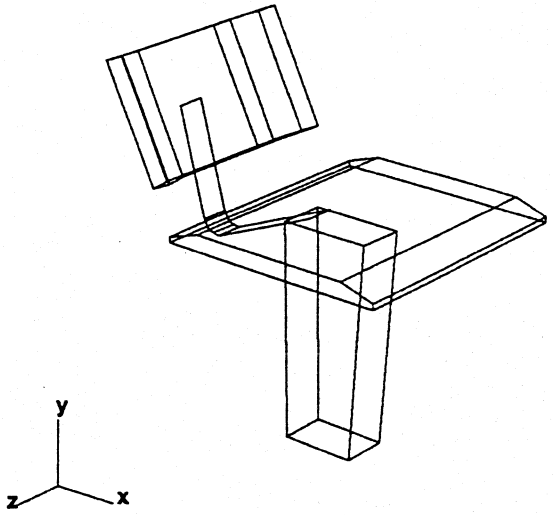
Example:

```
as
3
*** chair example
4
tz    seat support
0  0  0    0  0  0
5  4  -4
6         0
3         -40
-10       -40
-10       0
6         0
tz        seat
0  0  0    0  0  0
7  22  -22
16        3
22        1
22        0
-18       0
-18       1
-12       3
16        3
tz        back-rest support
0  0  0    0  0  0
6  2.5  -2.5
4         0
-17       3
-18       -2.5
-19       -2
-19.5     0
-19       20
ty        back-rest
-19  20  0    0  0  10
9  11  -11
8         -18
4         -11
4         11
8         18
4         18
0         11
0         -11
4         -18
8         -18
.................
. 2 other objects .
.................
```



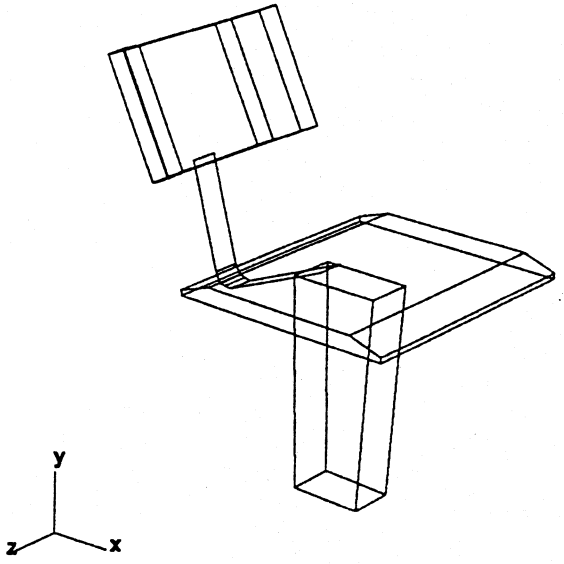
Changing the shift and rotation of the **back-rest support** gives:

```
as
3
*** chair example
4
tz          seat support
┌──────────────────────┐
└──────────────────────┘
tz            seat
┌──────────────────────┐
│                      │
└──────────────────────┘
tz back-rest support
-10  0  0   0  0  15
┌──────────────────────┐
│                      │
└──────────────────────┘
ty      back-rest
┌──────────────────────┐
└──────────────────────┘
.....................
.  other object     .
.....................
.  other object     .
.....................
```



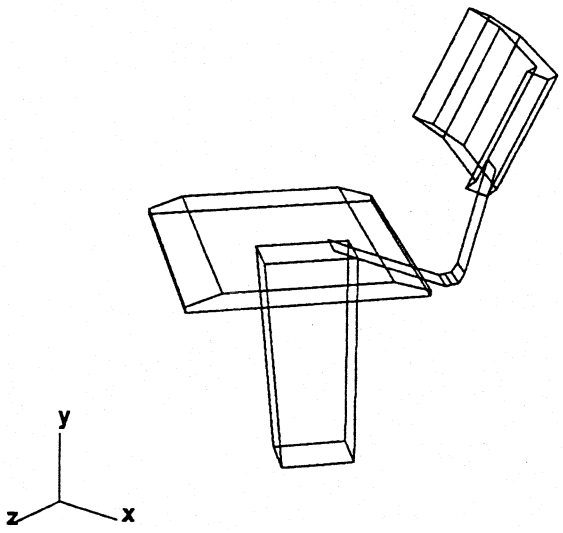
Shifting the **back-rest** :

```
as
3
**** chair example **
4
tz          seat support
|                       |
|_______________________|
tz                  seat
|                       |
|_______________________|
tz     back-rest support
|                       |
|_______________________|
ty              back-rest
-19  30  0    0  0  10   |
|                       |
|_______________________|
.....................
.  other object     .
.....................
.  other object     .
.....................
```



We end with a rotation of the whole chair: we just rotate the seat support, and the rest of the chair will follow:

```
as
3
**** chair example **
4
tz          seat support
0  0  0    0  -130  0
|_____________________|
tz          seat
|_____________________|
tz back-rest support
|_____________________|
ty     back-rest
|_____________________|
.....................
.   other object    .
.....................
.   other object    .
.....................
```



2.3 grid

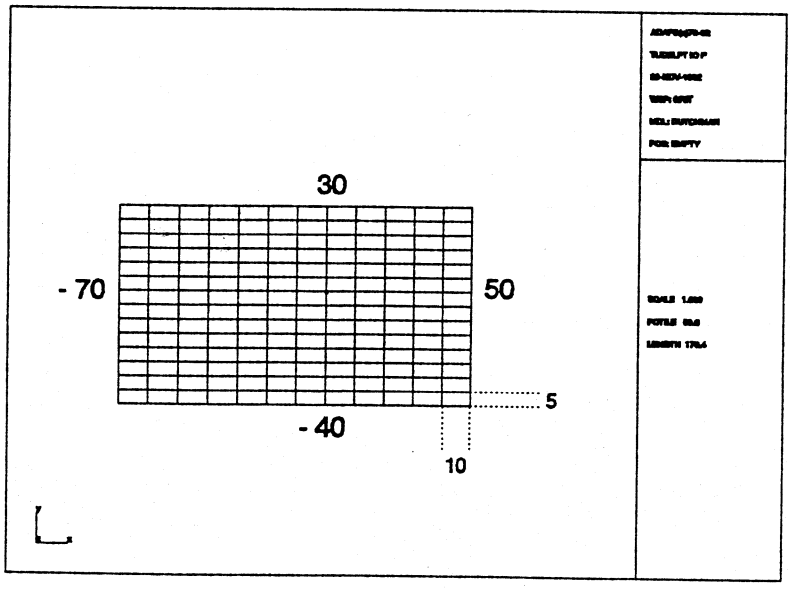
Next to the two description-modes for the workspace we can incorporate the following visual help: a grid, that we can define to our own liking and use as a fixed reference for measurement purposes.

Note: the grid option allows you to define **your own** grid as a fixed visual help. You can define, shift and rotate it, according to your own specific requirements. This grid is not the grid we get when we turn on the GRID-option in the ADAPS main-menu: that grid has a fixed mesh-width (10x10 cm when SCALE=1), and always covers the whole display.

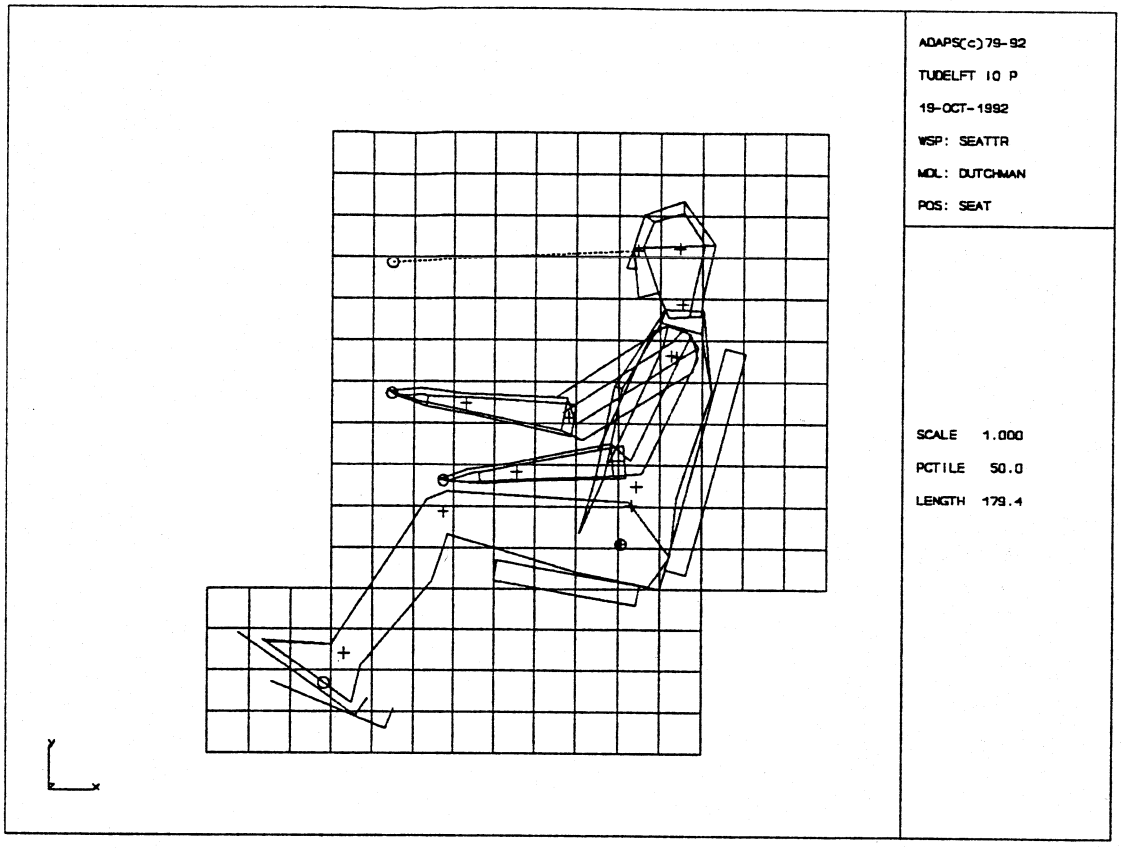
Example:

```
gr                                  code for grid; no leading blanks
0  0  0   0  0  0                   no shift/rotation C_i relative to C_o
-70  50  -40  30  10  5
```

```
└──────────────────────   xl: low  limit in X-dir. ; in cm
   └──────────────────    xh: high limit   ..      ;    ..
      └─────────────      yl: low  limit in Y-dir. ;    ..
         └──────────      yh: high limit   ..      ;    ..
            └─────        mx: mesh-width in X-dir. ;    ..
               └──        my:    ..         Y-  .. ;    ..
```



The next illustration shows an example of two grids for possible evaluation purposes:



Remember that you can place your grid-definition anywhere in the workspace description file:

as
gr
gr
ls

or

ls
as
gr

as long as you don't mix it: you cannot treat it as a graphical element inside an assembly definition.

2.4 Before we start modelling

It may be good to remind ourselves of the purpose of programs like ADAPS: they should aid in the assessment of human workspace interaction. Although now sitting in front of a computer-display, the design and evaluation of workspaces is still done by human users, the real experts in ergonomics or human factors.
No matter what level of graphics sophistication there exist on the computer-display, it still remains for the user to decide e.g. about the **kind of interaction** that will be relevant for a specific human-workspace design.

And so we should answer two basic questions:

1. What is our target population ?
   (Who do we design for ?).

2. What do we want them to do ?
   (What functional postures (including resting, doing nothing or moving around), are indicative for the interaction between the human(s) and the workspace ?).

In general, this results in a number of requirements that should accompany the design (and adjustment) of workspaces to human needs and capabilities:

- the human body should have free space; in rest, but also when moving;
- body-support where (and when) needed (this might conflict with the previous requirement);
- controls should be accessible (same remark);
- displays in field of view;
- sight on the surroundings or outside world.

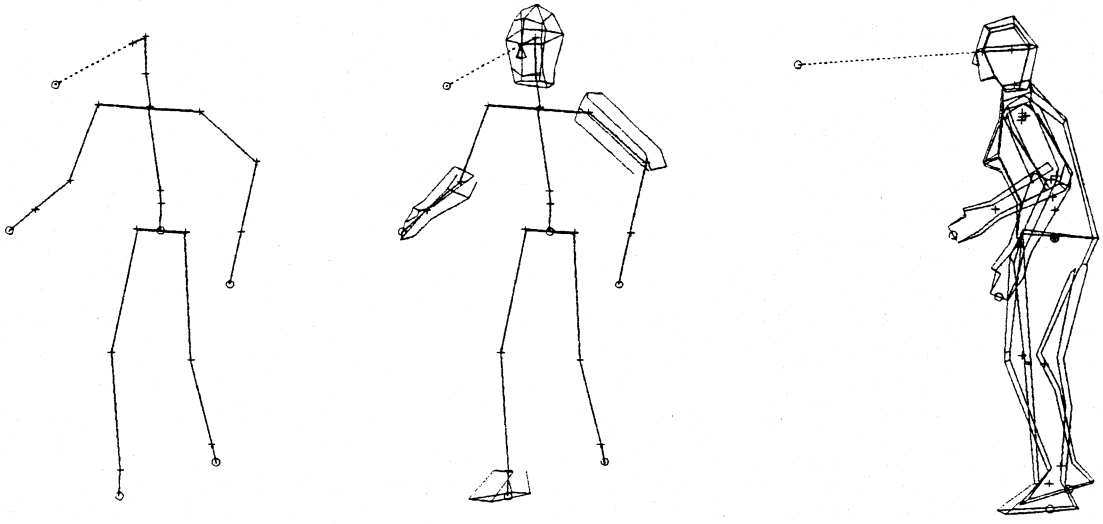
Translated into modelling guidelines:

- **keep it simple** (1) : we have wire-frame modelling, no hidden-line removal, so too many lines may easily clutter the display;
- **keep it simple** (2) : since the anthropometry of the human model is essentially simple; keep in mind that these models are the yardsticks we use to evaluate the workspace-design;
- **keep it simple** (3) : only model those parts of the workspace design (supports, controls, displays, field-of-view restrictions (windows) etc.), that are essential for understanding the interaction with functional postures of the human model; details of workspace-construction are not important in this stage;
- model **functionally** (see the chair example) and **efficiently** (it's easy to make copies of a workspace-file; use those to study design alternatives).

3. The Model

An ADAPS-model consists of a set of linear branched chains, containing twentyfive **links** or body-members (see figure below-left).

Relative to these links we define **surface-points** that, together with a number of **lines** between these points, determine the outside geometry of the model (middle and right figure).

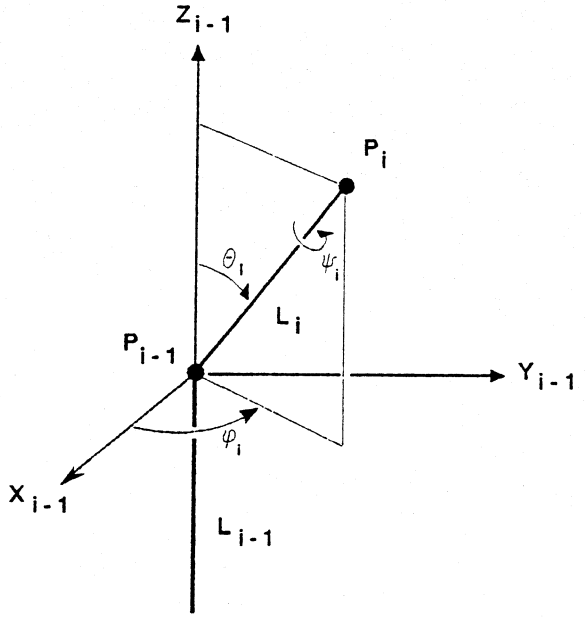


A link's oriëntation is defined by its **joint-angles**, relative to the connecting, more proximal link. Since the surface-points are defined relative to the links, changing the oriëntation of a link will also change the position of the surface-points in space (while keeping their relative position to the link) and in this way the model's outside shape.

Thus: a set of **links**, their oriëntation via **joint-angles**, the coordinates of **surface-points** relative to the links, and a number of **lines** between the surface-points, are all that is needed to (re)create an ADAPS-model.

Link-oriëntation

The oriëntation of link L_i , relative to the connecting, more proximal link L_{i-1} , is defined via the three joint-angles φ_i , θ_i and ψ_i .
These angles are measured in the local co-ordinate system C_{i-1} of L_{i-1} , situated at the distal end P_{i-1} of L_{i-1} (see the figure below):



φ_i determines the plane in which L_i bends when changing θ_i ;
ψ_i determines axial rotation of L_i .

Conventions:

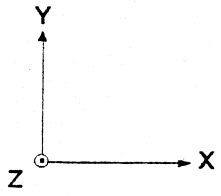
φ_i is positive [negative], starting from de X_{i-1} -axis, in the direction (along the smallest angle) of the positive [negative] Y_{i-1} -axis.

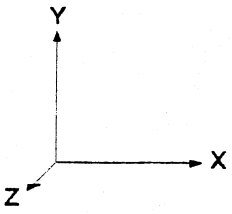
φ_i = 90 --- θ_i is positive [negative] : L_i bends towards the positive [negative] Y_{i-1} -axis

φ_i = -90 --- θ_i is positive [negative] : L_i bends towards the negative [positive] Y_{i-1} -axis

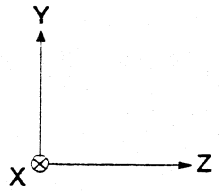
ψ_i is positive for a clockwise rotation, seen from P_{i-1} towards P_i .

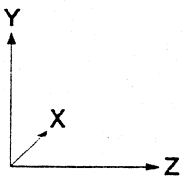
In the next page you will find an overview of the model's local co-ordinate systems. In the overview we have the next conventions:





means: Z -axis out of the plane of the paper (towards the viewer)



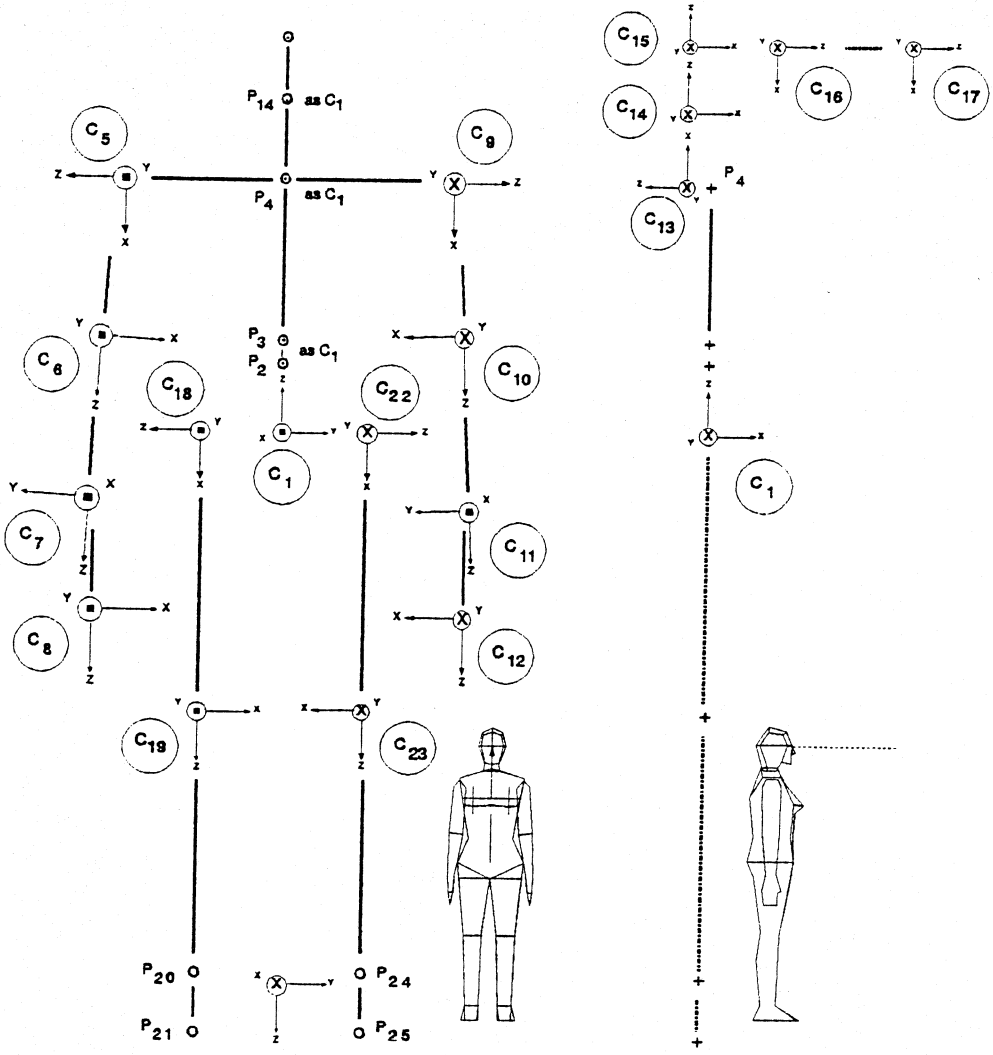


means: X -axis into the plane of the paper (away from the viewer)

The four pages after that, display the data for the joint-angles (start values and range) for rotations in the frontal, sagittal and transverse plane.

At the end of the manual you will find a two-page overview.

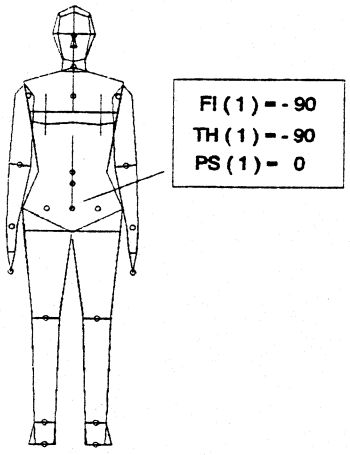
schematic display of the model's local co-ordinate systems



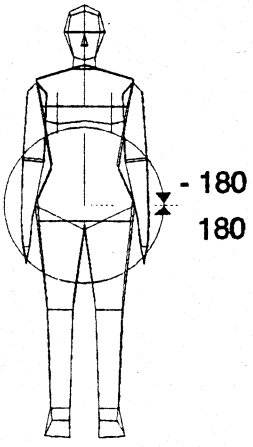
front view

side view

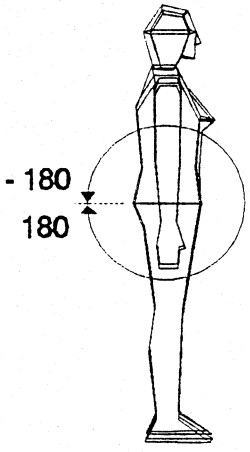
BASEPOINT (WHOLE MODEL) ROTATION : START VALUES AND RANGE



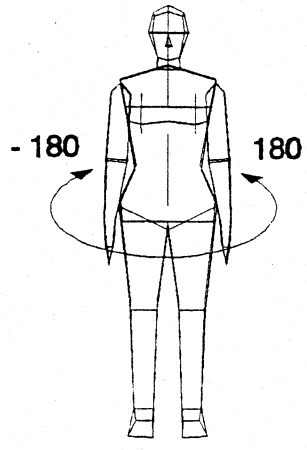
START POSITION : FRONT VIEW OF MODEL



FI (1) = - 90



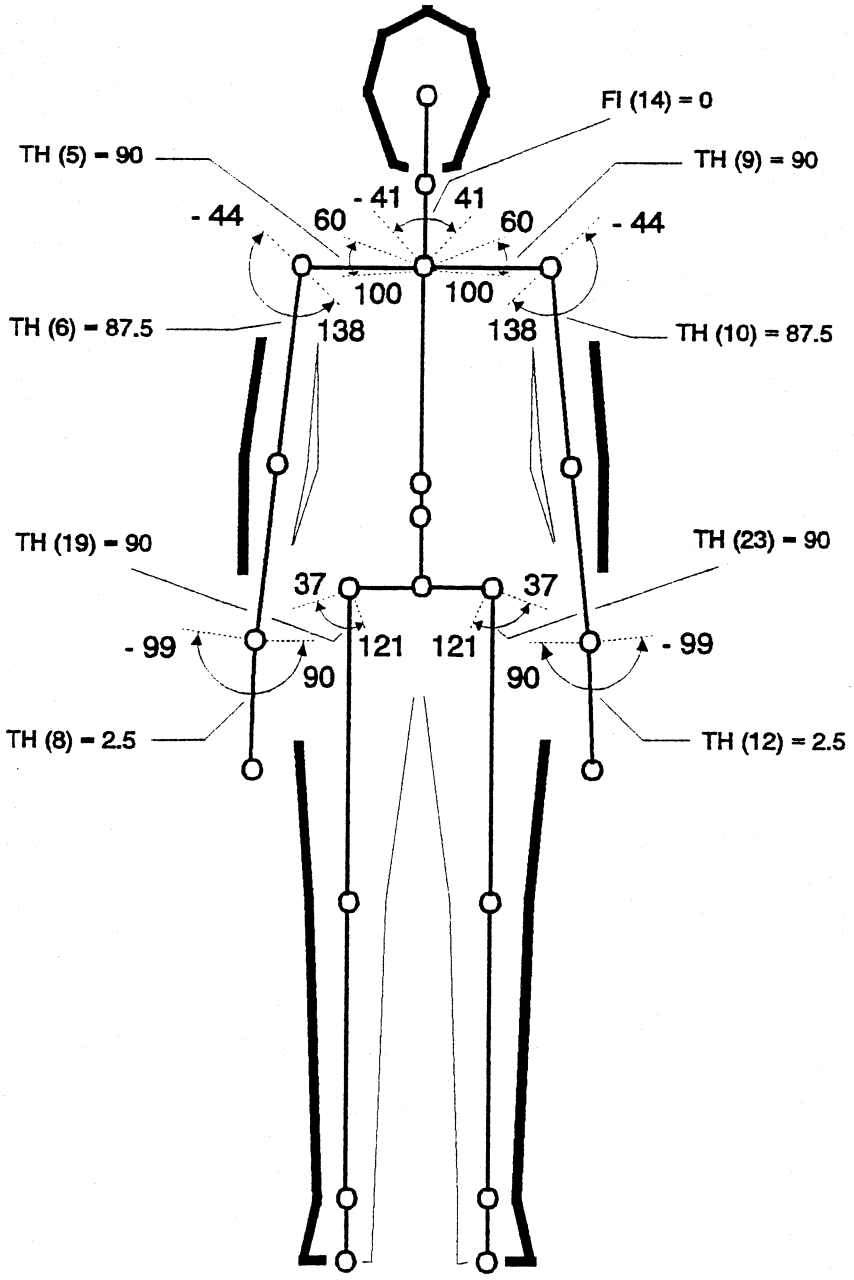
TH (1) = - 90

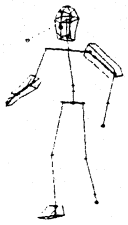


PS (1) = 0

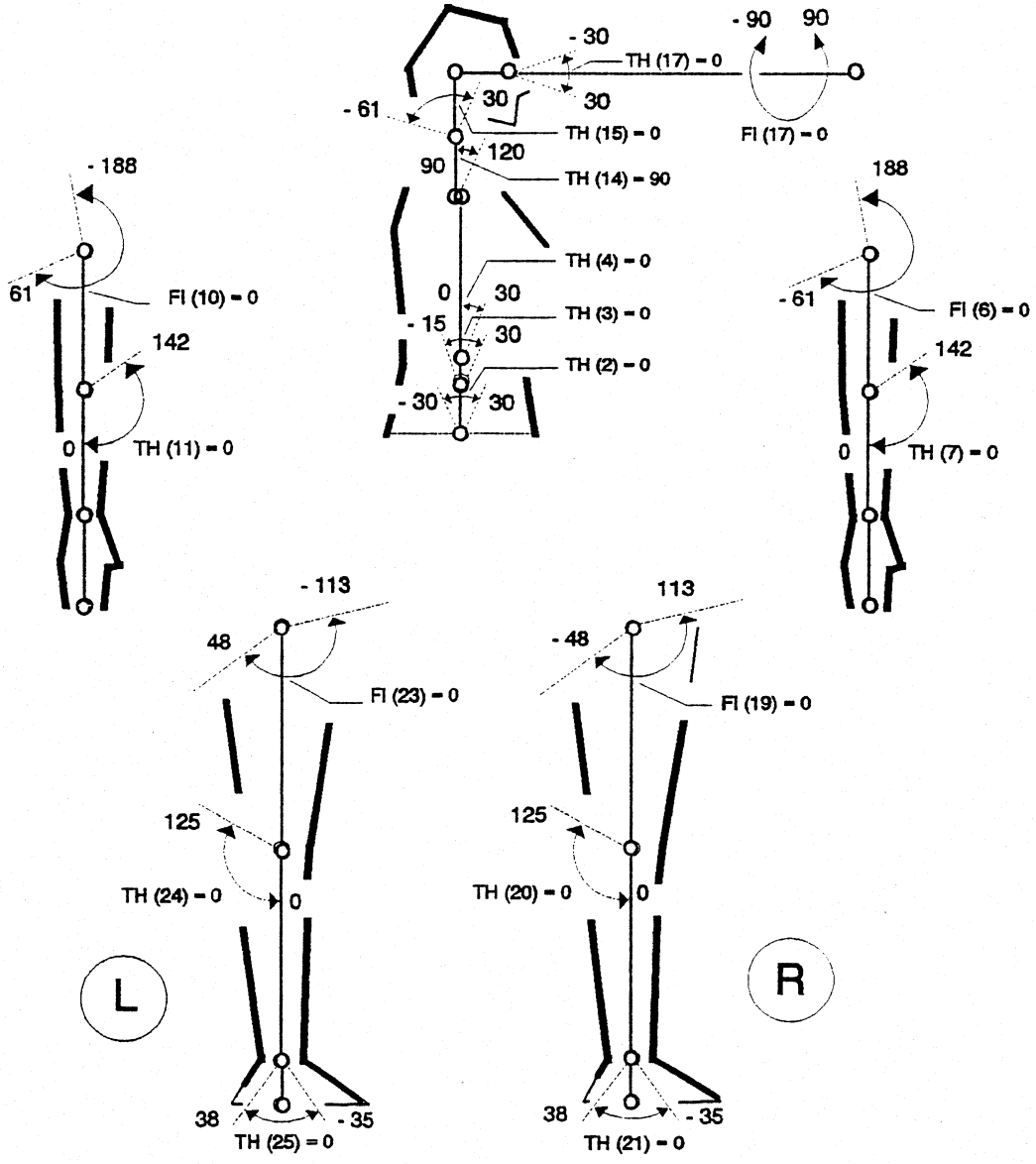
ROTATIONS IN FRONTAL PLANE : START VALUES AND RANGE

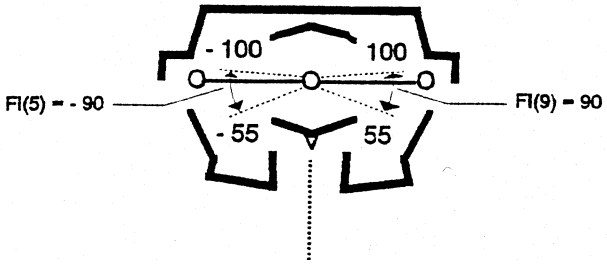
FRONT VIEW





ROTATIONS IN SAGITTAL PLANE : START VALUES AND RANGE

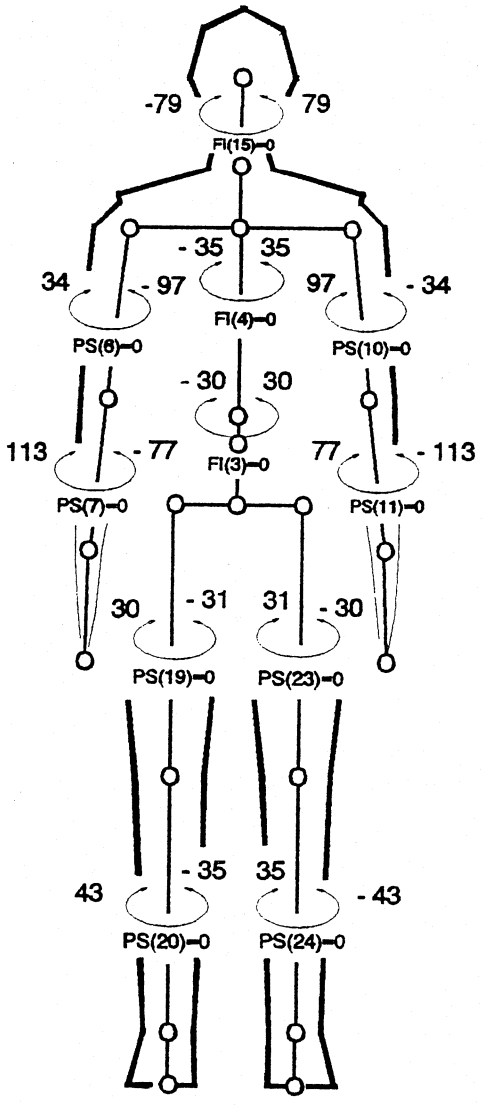




TOP VIEW

ROTATIONS IN TRANSVERSE PLANE : STARTVALUES AND RANGE

FRONT VIEW



Model-files

A model's links (their name and lengths) and the co-ordinates of the surface-points, are stored in a (formatted) ASCII-file with the extension **.MDL** .

The joint-angles (start-values and range) and the code for the lines between the surface-points are stored in a (formatted) ASCII-file with the extension **.ANG** .

To (re)create e.g. the model for the Dutch female, we need the files **DUTCHWMN.MDL** and **DUTCHWMN.ANG** .

Since the model-files are ASCII-files, they can be adapted (e.g. you might want to change the range for the joint-angles or create your own, special-purpose model), as long as you remember that the files are formatted-files: the place of the data in the **.MDL** and **.ANG** files is relevant. For a complete description of the ADAPS models see the relevant literature cited below.

At this moment the following human-models are available:

"American Pilot" - files: **AMPILOT.MDL** and **AMPILOT.ANG** ; the first ADAPS-model (see e.g. Post, 1979 and Hoekstra,1985a); base: Flying Personnel-1950, (Hertzberg et al., 1954) ;

"4-year old boy" - files: **BOY04.MDL** and **BOY04.ANG** ; (Hoekstra, 1986); based on available data (de Rijke,1985); to be replaced in the future with separate models for the 2-12 year range;

"Dutch male" - files: **DUTCHMAN.MDL** and **DUTCHMAN.ANG** ; (Hoekstra, 1985c); based on available Dutch data and estimates: DINEDII (Molenbroek, Dirken, 1984);

"Dutch female" - files: **DUTCHWMN.MDL** and **DUTCHWMN.ANG** ; (Hoekstra, 1985b); based on available Dutch data and estimates: DINEDII (Molenbroek, Dirken, 1984);

"Dutch elderly male" - files: **ELDERMAN.MDL** and **ELDERMAN.ANG** ; (Ruiter, 1993b); based on data about Dutch elderly: (Molenbroek et al., 1983);

"Dutch elderly female" - files: **ELDERWMN.MDL** and **ELDERWMN.ANG** ; (Ruiter, 1993a); based on data about Dutch elderly: (Molenbroek et al., 1983);

Literature

Hertzberg et al. (1954) — Hertzberg, H.T.E., Daniels, G.S., Churchill, E. Anthropometry of Flying Personnel-1950, Technical Report WADC TR 52-321, Wright Air Development Center, Wright-Patterson Air Force Base, Ohio, september 1954, (AD 47 953)

Hoekstra (1985a) — Hoekstra, P.N. ADAPS - Herkomst van de gegevens van het antropometrisch model; TH-Delft, Tussenafdeling Industriëel Ontwerpen, Vakgroep Produktergonomie, maart 1985

Hoekstra (1985b) — Hoekstra, P.N. ADAPS-WOM84 : voorlopig antropometrisch model van de Nederlandse vrouw; TH-Delft, Tussenafdeling Industriëel Ontwerpen, Vakgroep Produktergonomie, juni 1985

Hoekstra (1985c) — Hoekstra, P.N. ADAPS - MAN84 : voorlopig antropometrisch model van de Nederlandse man; TH-Delft, Tussenafdeling Industriëel Ontwerpen, Vakgroep Produktergonomie, juli 1985

Hoekstra (1986) — Hoekstra, P.N. ADAPS - BOY04 : voorlopig antropometrisch model van de 4-jarige Nederlandse jongen; TU-Delft, Faculteit van het Industriëel Ontwerpen, december 1986

Levis et al. (1980) — Levis, J.A., Smith, J.P., Porter, J.M. and Case, K.; The impact of Computer-aided Design on Pre-concept Package Design and Evaluation.In: Human factors in Transport Research. Eds. D.J.Oborne and J.A. Levis. 1980 Vol.1, 356-364. Academic Press, London

Molenbroek et al. (1983) — Molenbroek, J.F.M., Houtkamp, J.J., Burger, A.K.C. Bejaardenantropometrie; TU-Delft, Faculteit van het Industriëel Ontwerpen, reeks 'IO-bijzondere onderwerpen', deel 6, 1983

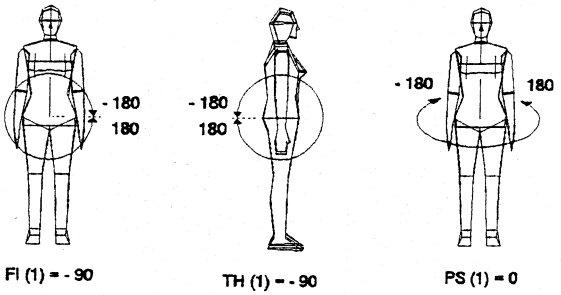
x2

Molenbroek en Dirken (1984)	Molenbroek, J.F.M., Dirken, J.M. Nederlandse lichaamsmaten voor Ontwerpen; DINED-tabel (2e herziene versie), TH-Delft, Tussenafdeling Industriëel Ontwerpen, Vakgroep Produktergonomie, oktober 1984
Post (1979)	Post, F.H. Werkplek-ontwerpen met antropometrische en biomechanische computer-modellen; 2 delen, TH-Delft, Tussenafdeling Industriëel Ontwerpen, Vakgroep Produktergonomie, mei 1979
de Rijke (1985)	de Rijke, M.M. Kinderen en hun maatjes - bepaling van lichaamsmaten van Nederlandse kinderen ten behoeve van onderzoek naar speeltoestellen; TH-Delft, Afdeling van het Industriëel ontwerpen, AB 807, augustus 1985
Ruiter (1993a)	Ruiter, I.A. ADAPS - ELDERWMN Voorlopig antropometrisch model van de bejaarde Nederlandse vrouw; TU-Delft, Faculteit van het Industriëel Ontwerpen, Vakgroep Produktergonomie,januari 1993
Ruiter (1993b)	Ruiter, I.A. ADAPS - ELDERMAN Voorlopig antropometrisch model van de bejaarde Nederlandse man; TU-Delft, Faculteit van het Industriëel Ontwerpen, Vakgroep Produktergonomie,januari 1993

For a recent overview see:

Karwowski (1990)	Karwowski, W., Genaidy, A.M., and Asfour, S.S., Computer-Aided Ergonomics - A Researchers Guide, Taylor & Francis, London, 1990

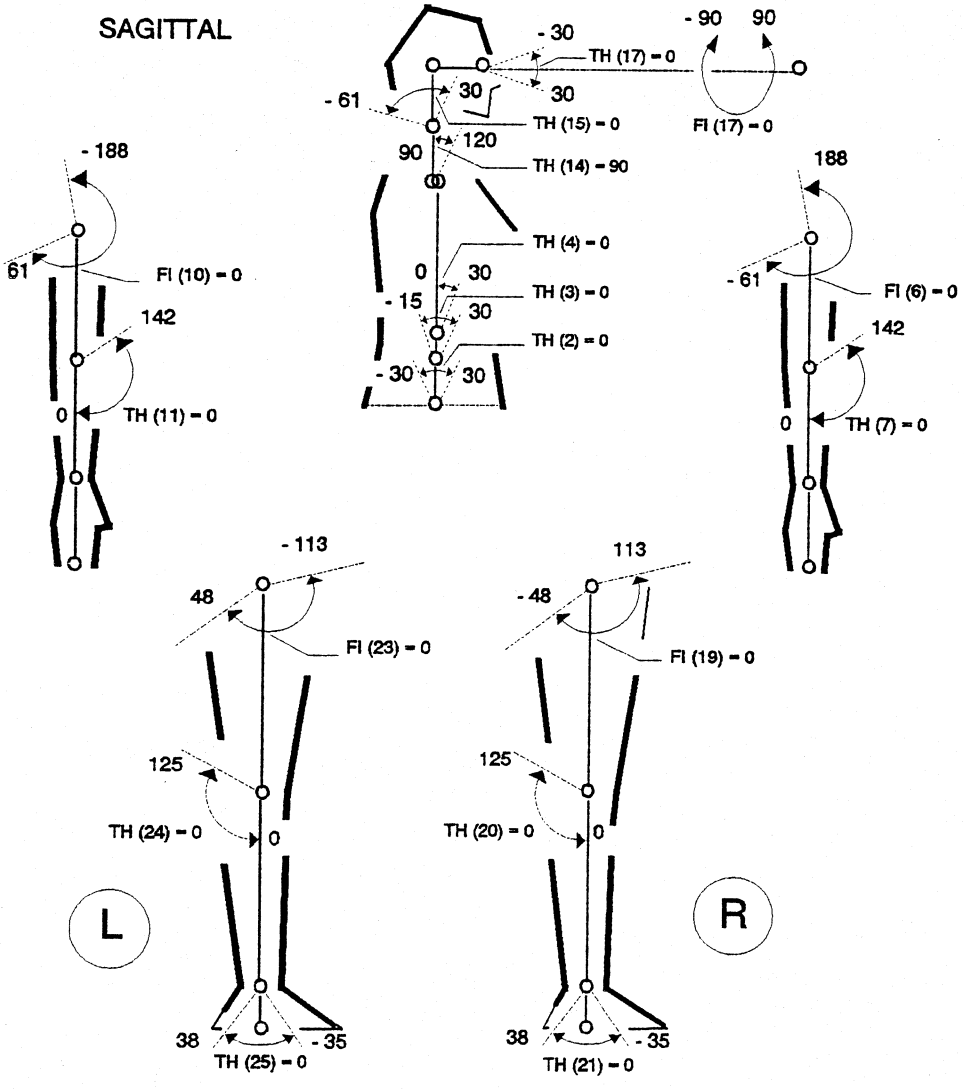
WHOLE MODEL



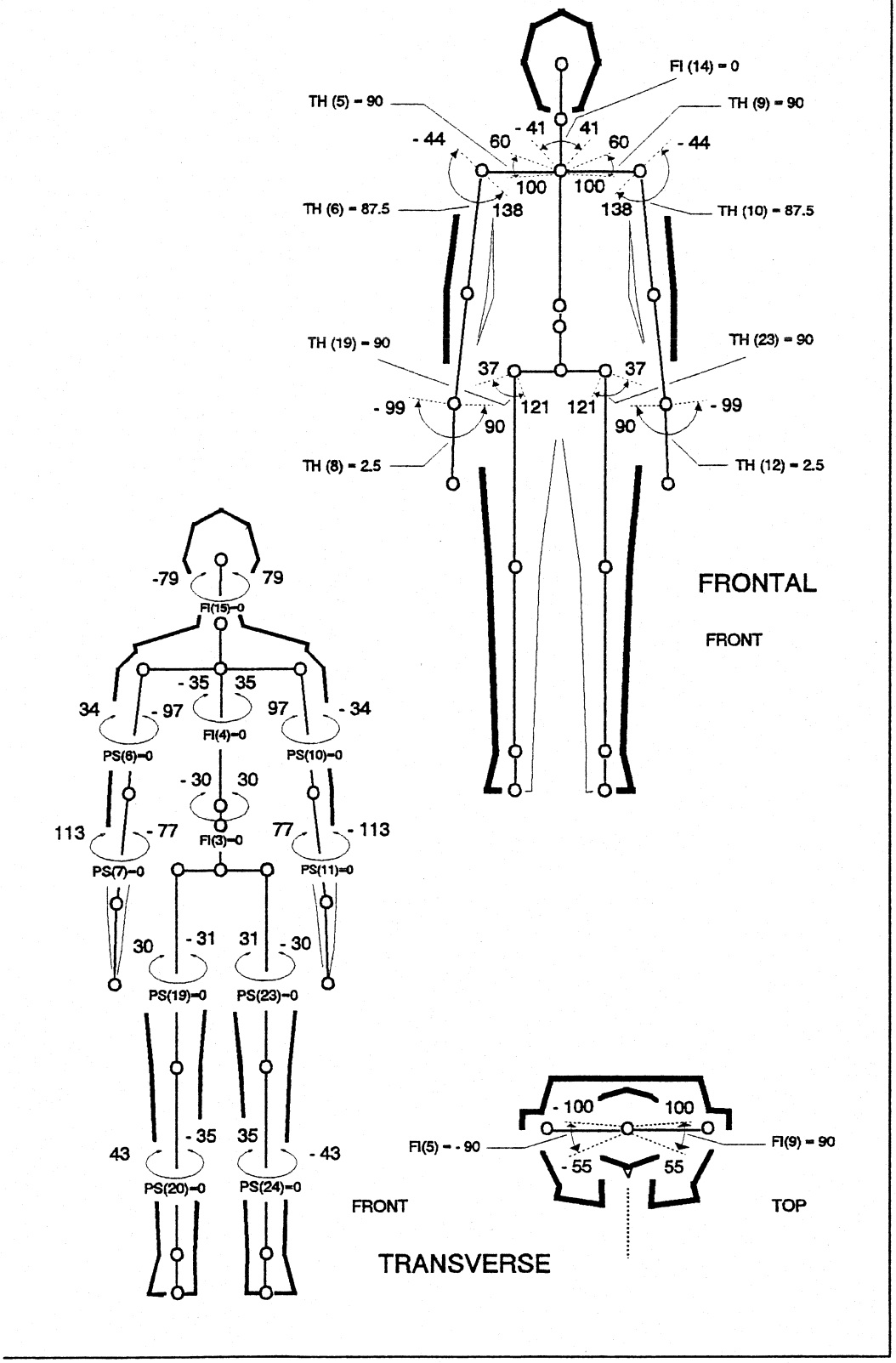
ADAPS
JOINT - ANGLES

START VALUES
AND RANGE

SAGITTAL



x4


Fl (14) = 0
TH (5) = 90
TH (9) = 90
- 44
60
- 41
41
60
- 44
100
100
TH (6) = 87.5
138
138
TH (10) = 87.5
TH (19) = 90
TH (23) = 90
37
37
- 99
121
121
- 99
90
90
TH (8) = 2.5
TH (12) = 2.5
FRONTAL
FRONT
-79
79
Fl(15)=0
- 35
35
34
- 97
97
- 34
Fl(4)=0
PS(6)=0
PS(10)=0
- 30
30
113
- 77
77
- 113
Fl(3)=0
PS(7)=0
PS(11)=0
30
- 31
31
- 30
PS(19)=0
PS(23)=0
- 35
35
43
- 43
PS(20)=0
PS(24)=0
- 100
100
Fl(5) = - 90
Fl(9) = 90
- 55
55
FRONT
TOP
TRANSVERSE