FACE TRACKING NAO

FUNCTIONALITY:

This library allows the robot to be able to follow a face with the head.

Therefore, the general functioning of the library can be divided into these two stages:

• Face Detection.

• Follow the face with the head (Face Tracking).

1. FACE DETECTION:

This stage is done with the help of OpenCV, which works using 'cascades'. These 'cascades' are based on detecting faces in multiple stages. For each block an approximate and fast test is made, and if this test is passed, it makes a little more detailed test, and so on. The algorithm can have 30-50 of these stages or cascades, and will only detect one face if all stages pass.

Cascades are just a bunch of XML files that contain OpenCV data used to detect objects, so you just need to initialize the code with the cascade you want and then this already does the job. In this case the cascade used is of the Haar type and serves to recognize faces that are facing the image.

An example of how this algorithm for face detection works is shown in figure 1. Starting from the image shown at the top and adjusting a series of parameters you can get the result shown at the bottom of the same figure.

Among the parameters to be adjusted is the scale factor, used to compensate for the fact that some faces may be closer to the camera than others. Other factors have to do with the fact that the detection algorithm uses a moving window to detect objects, so MinNeighbors defines how many objects are detected near the current window before declaring the face found, while MinSize gives the size of each window.

The algorithm returns a list of rectangles where it thinks it found a face, which are defined by four values that are: the location x and y of the rectangle, and the width and height of the rectangle (w, h). Therefore these are also used to draw the rectangles that enclose the faces and thus to be able to verify that the algorithm is working well. On the other hand, from the width and height it is determined which is the largest face, which will be the one that will be followed in case the robot sees more than one at a time.

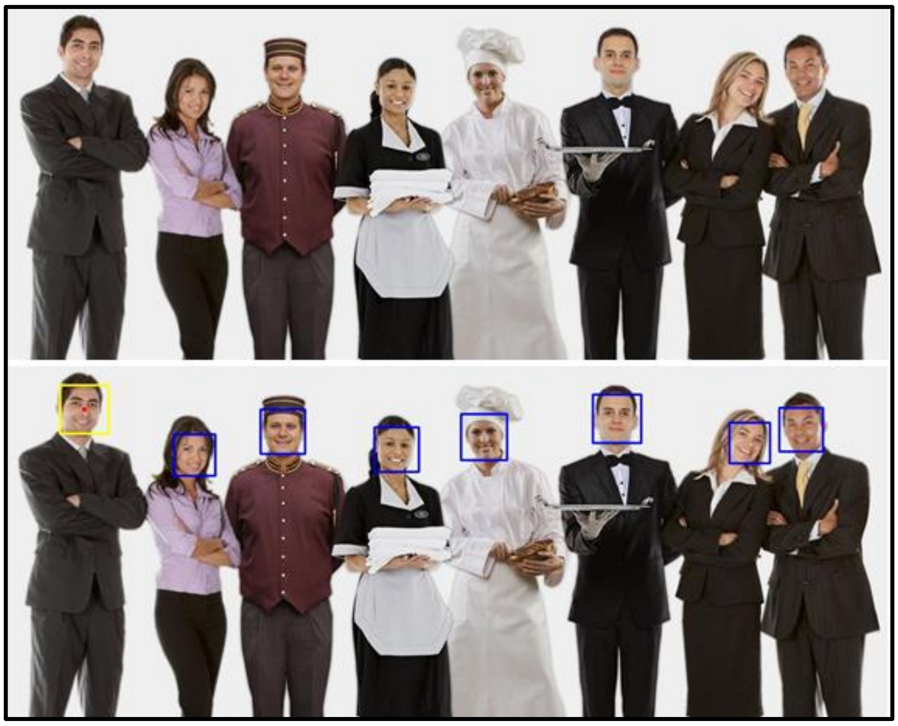


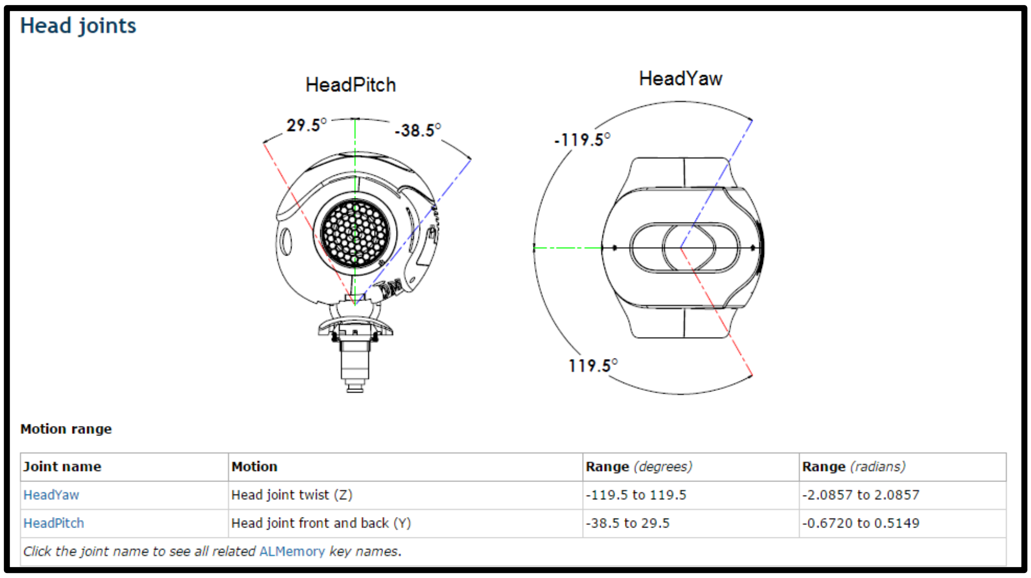
Figure 1 – Example of face detection

.

2. FACE TRACKING:

- **Head movements.**

The head movement necessary to track the detected face is calculated so that the ball is always kept in the center of the image being obtained in real time through the camera of the Nao robot



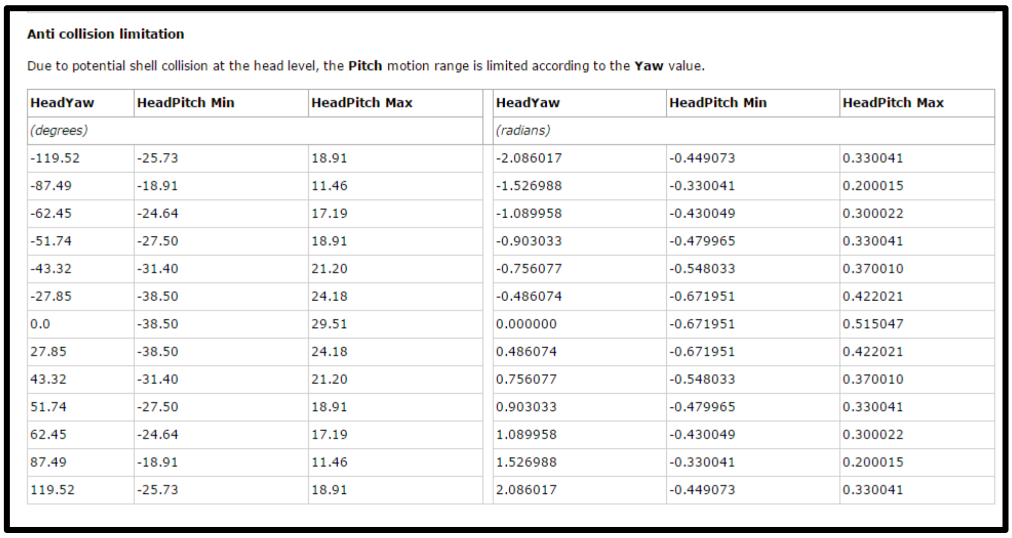
Motion range

Joint name	**Motion**	**Range** *(degrees)*	**Range** *(radians)*
HeadYaw	Head joint twist (Z)	-119.5 to 119.5	-2.0857 to 2.0857
HeadPitch	Head joint front and back (Y)	-38.5 to 29.5	-0.6720 to 0.5149

Click the joint name to see all related ALMemory key names.

Figure 2 - Head movement ranges.

Some movements of the head are restricted, figure 3, to prevent this one from colliding with the shoulders of the robot and to avoid its wear. For this, in each iteration is made a prediction of which will be the new position of the head and, if this is conflicting, its movement is restricted to avoid the shock.



Anti collision limitation

Due to potential shell collision at the head level, the **Pitch** motion range is limited according to the **Yaw** value.

HeadYaw	**HeadPitch Min**	**HeadPitch Max**	**HeadYaw**	**HeadPitch Min**	**HeadPitch Max**
(degrees)			*(radians)*		
-119.52	-25.73	18.91	-2.086017	-0.449073	0.330041
-87.49	-18.91	11.46	-1.526988	-0.330041	0.200015
-62.45	-24.64	17.19	-1.089958	-0.430049	0.300022
-51.74	-27.50	18.91	-0.903033	-0.479965	0.330041
-43.32	-31.40	21.20	-0.756077	-0.548033	0.370010
-27.85	-38.50	24.18	-0.486074	-0.671951	0.422021
0.0	-38.50	29.51	0.000000	-0.671951	0.515047
27.85	-38.50	24.18	0.486074	-0.671951	0.422021
43.32	-31.40	21.20	0.756077	-0.548033	0.370010
51.74	-27.50	18.91	0.903033	-0.479965	0.330041
62.45	-24.64	17.19	1.089958	-0.430049	0.300022
87.49	-18.91	11.46	1.526988	-0.330041	0.200015
119.52	-25.73	18.91	2.086017	-0.449073	0.330041

Figure 3 - Restriction of head movements.