Fast Posture and Object Recognition using Symmetries

Nathan Lovell
Griffith University, Australia
nathan.lovell@student.gu.edu.au

Abstract

We present a fast technique for the task of posture recognition that makes use of the inherent symmetries found in the geometries of objects to determine the pose relative to the viewer. Our technique is very versatile. Not only will it correctly determine the relative posture of viewed objects, but it will do so in a way that is independent of the distance between the object and the viewer. We show that our technique is useful in a wide variety of problem domains by illustrating it in three very different recognition applications. Furthermore our technique is shown to be fast enough to use in a real-time robotics application (on the Sony AIBO) where processing power is limited.

1 Introduction

Many techniques are available in the literature for recognising the posture of a known object. Much of this work focuses on recognition of facial or hand gestures as there are immediate applications of this in many areas including user interface design, face recognition and security monitoring systems. This work tends to be very domain specific. For example, recognising a facial gesture (posture) relies on domain knowledge such as how to locate the eyes and nose within an image [5]. It is not our contention that these tasks could be made simpler - in fact, it is hard to see how you could perform these tasks without the computational expense associated with the extraction of complex features such as the eyes and nose. We postulate that because these techniques are so well used and known, many simpler methods get overlooked when the recognition task is itself more simple. For example, we will show that it is not necessary to extract the features of a coffee cup handle to detect the pose of a coffee cup relative to the viewer. This task can be performed at a lower level.

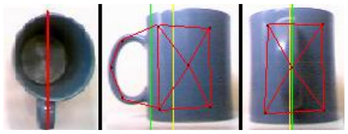
Even at a high level, posture recognition can often be simplified by performing a low-level analysis before high level feature extraction is performed. One technique for performing exactly the same task as our main example - recognising the postures of Sony AIBO robots for the task of playing robot soccer - relies on learning the shapes of coloured patches on the soccer uniform of the AIBO [4; 14]. There are other techniques for posture detection as well. Some are based on perimeters of objects [12] and some on the comparison of surfaces to other planar surfaces (circles, squares etc.) [5]. Each of these techniques has in common a complex preprocessing phase that must be performed in order to extract the necessary features (usually lines and edges). Some tasks simply do require high-level processing. Others, however, benefit from a lower-level posture recognition algorithm. Shapes on an AIBO's soccer uniform, for example, are much more readily identified by symmetry than by edge analysis and the posture of an AIBO can be determined much more quickly this way. Simple hand gestures too, are more inexpensively identified by symmetry than by finding features such as knuckles and fingers.

We present here several examples of posture recognition using the symmetries of real world objects and we apply our technique in several complex visual domains. We first illustrate our technique using the tasks of analysing the posture of opponent robots in a game of robotic soccer. Secondly we will perform posture recognition of hand gestures and finally we apply our method to the analysis of maritime signaling flags. Even though these are widely varied domains, each lend themselves to our symmetry analysis approach to object recognition.

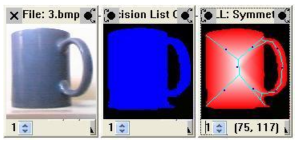
2 Object Symmetries

Most real world objects have at least one line of symmetry. Our technique can use one or more lines of symmetry, however, we have found it to be particularly successful in the case when we identify the posture of objects that have exactly one symmetrical axis.

For example, consider a common coffee mug as in Figure 1 (a). When viewed from the top the mug has a



(a) The symmetry of a coffee mug. M_{S_x} is shown in green, M_{B_x} is shown in yellow.



(b) Medial-axis analysis to form the skeleton of the mug. First find the blob of the cup, then identify the medial axis.

Figure 1: Coffee cup analysis.

single line of symmetry passing through the centre of its handle. If the mug is viewed from directly in front (that is, with the handle facing us) or from directly behind (with the handle behind the cup) then the cup will look symmetrical to us. If, however, we view the cup from the side then the handle will create an asymmetry which we can use to identify the posture of the object.

We use a technique very similar to the computation of the medial axis [3] to identify the structure of an object[1]. We define the skeleton of the object as the set of all locally maximal pixels in the medial axis. This skeleton can then be easily analysed for symmetry using the techniques presented below. A symmetrical skeleton would indicate that the object is being viewed from either in front or behind. The further from symmetry the skeleton is, the larger the rotation of our viewpoint relative to the object is (refer again to Figure 1 (a)).

The first step when computing the skeleton is to segment the image to identify coloured regions that belong to our object. Colour segmentation is an extremely common task in visual processing systems so our approach has no extra overhead compared to other systems because of this step. We have previously described a method for efficiently doing this first step [7].

After the areas of the image (we will refer to them as blobs) that correspond to the object are identified, we can find the medial axis by stripping pixels from the edge of the blob until the medial axis is evident [3] (see Figure 1 (b)). We have presented several improvements on this technique, in particular a fast method for computing this skeleton from a poly-line representation of the outline of the blob [7] and a linear time algorithm for computing the poly-line representation given the blob [10]. Applying the methods in our previous work results in an algorithm that is an order of magnitude faster than the original stripping technique because not every pixel needs to be examined.

The final step is to identify the skeleton, S, from the medial axis, A. We care only about the nodes of the skeleton - the lines are drawn on the images in the figures to aid human understanding. To find S we assign each pixel in the medial axis $p \in A$ a support value V_p based on how deep it is within the bounds of the blob (that is, the lighter areas in Figure 1 (b) are assigned higher weights). The local maxima within A are found by comparing each value V_p with the values of the other pixels in A that are close to p. If a unique local maxima is found within a given area then that pixel is added to S. However, sometimes there will be a number of pixels in an area with the same support value. In this case, our algorithm selects one pixel as a representative and adds that to S. By placing the nodes V_p in a heap data structure, the entire process is achieved in linear time relative to the number of pixels in the blob.

2.1 Symmetries in the Medial Axis

Using skeletons for the task of object recognition is common [11; 6; 7]. Usually, after the medial axis is found a match is sought between the medial axis and some learned or stored pattern. Our new technique also differs at this point.

The medial axis is analysed for symmetry by comparing its median point to the median point of the entire colour-segmented blob. This is a simple task that can be performed efficiently in one or more dimensions[2]. The median point on the skeleton is calculated from the set of all points in the skeleton while the median point of the blob is calculated from the set of all pixels in the blob. We refer to the median points as $M_S = (M_{S_x}, M_{S_y})$ or

[1]We define the medial axis slightly differently to Chin, Snoeyink and Wang who define the medial axis of a polygon as the boundary of the Voronoi diagram of the edges of the polygon. Our technique requires more information than this, but is an identical transform. We define the medial axis of a polygon as the raster representation of that polygon where every pixel inside the polygon is assigned a weight equal to its Manhattan distance to the nearest edge of the polygon. The set of pixels with locally maximal weights will be identical to the medial axis defined in [3].

[2]Strictly speaking the median point in 2 dimensions of the set $Q = \{(x_1, y_1), ...(x_n, y_n)\}$ is the point $P = (x, y) \in Q$ that minimises $f(x) = \sum_{i=1}^{n} \parallel P - Q_i \parallel$ where $\parallel P - Q_i \parallel$ is the Euclidean distance between the points P and Q_i. Finding the spatial median is intractable [2] so this is not the definition we use. Rather we define $\parallel P - Q_i \parallel$ to be the *Manhattan* distance between P and Q_i. In this way the median point may be computed in linear time on the x and y components independently. We use medians rather than means because medians are more robust estimators of central tendency than means [13].

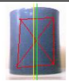
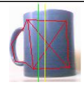
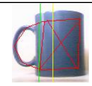
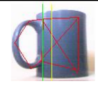
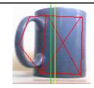
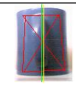
Actual Orientation	0^o	45^o	90^o	120^o	150^o	180^o
						
Perceived Orientation	-1^o	49^o	86^o	118^o	153^o	179^o

Figure 2: We determine the orientation of the coffee cup by the relation between M_{S_x} (green) and M_{B_x} (yellow).

M_B= (M_{B_x}, M_{B_y}). The median of all the x-projections for all points in the skeleton S is denoted as M_{S_x}. Similarly the median of all the y-projections for all points in the skeleton is M_{S_y}. In the same way the medians for the x and y projections of all points in the blob are M_{B_x} and M_{B_y}.

The relationship between M_S and M_B reveals the viewed objects orientation. Refer to Figure 2. This figure illustrates how the distance between M_{S_x} and M_{B_x} varies with the orientation of an object with one axis of symmetry. M_{S_x} is represented by a green line in these images, while M_{B_x} is the yellow line.

If the distance between M_{S_x} and M_{B_x} is large then the object is at its most non-symmetric orientation relative to the viewpoint. In fact, there is a direct correlation between the perceived angle of orientation (θ) that is proportional to the inverse sine of the distance between M_{S_x} and M_{B_x}. The correlation fits closely the equation

$$\theta = \arcsin(d/max_d). \qquad (1)$$

The variable max_d represents the distance in pixels between M_S and M_B at 90^o orientation. This calculation can be adjusted appropriately for objects with more than one axis of symmetry. Furthermore our technique is largely independent of the distance to the viewed object. The distance between the two median points, with respect to max_d, may be normalized against the overall size of the blob. This means that the computationally expensive process of image normalization, which is common in posture recognition systems [5], is not required for our technique to work. Normalization can be performed as the final step on several pre-calculated points, instead of the initial step on the entire image.

Equation (1) fits well when the viewing angle is in the range $-90^o < \theta < 90^o$. If the actual angle of orientation lies in the range $90^o < \theta \leq 180^o$ or $-90^o > \theta \geq -180^o$ then the solution provided by our equation will be out of phase. Consider Figure 3. For each orientation within $90^o < \theta \leq 180^o$ or $-90^o > \theta \geq -180^o$ there is another within $-90^o < \theta < 90^o$.

To discriminate between these possibilities we introduce a new feature - the bounding rectangle of the blob, R. M_R= (M_{R_x}, M_{R_y}) where M_{R_x} is the median of the x-projection of all pixels in R and M_{R_y} is the median of the y-projection of all pixels in R.

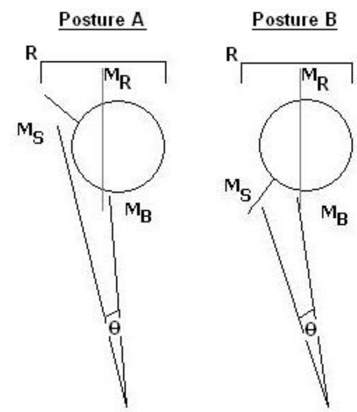


Figure 3: Two different postures that lead to the same perceived angle due to equal distance between M_{B_x} and M_{S_x}. We resolve this by comparing with M_{R_x}.

When our viewing angle is close to 0^o the object looks completely symmetric to us. There is a set of orientations that produce a skeleton homomorphic to the one produced at 0^o. Parts of the object viewed from this set we will label the FRONT of the object. The parts of the object that are only visible from outside this set of angles are labeled the SIDE. In our coffee cup example, we are viewing more of the SIDE of the object as the handle moves into view.

When θ is small, the perspective of the viewpoint ensures that the SIDE of the object will have few pixels in it compared to the FRONT. This is because it is further away from the viewpoint. We call any posture visible from the set of angles $-90^o < \theta < 90^o$, Posture A. The ratio of pixels in the SIDE to the FRONT of the object when viewed in Posture A is $Ratio_A$. Each possible Posture A, however, has a mirror posture shown in Figure 3. Both of these postures render the same angle θ using Equation (1) because $|\, M_{B_x} - M_{S_x}|$ is equal. We call the mirror Posture B and the ratio of pixels in the FRONT to the SIDE of the object in this posture $Ratio_B$. The perspective of the camera ensures that $Ratio_B$ will be

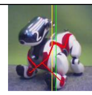
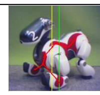
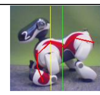
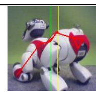
Actual	0^o	30^o	60^o	90^o
				
Perceived	-3^o	27^o	65^o	86^o
Actual	110^o	130^o	150^o	180^o
				
Perceived	103^o	121^o	145^o	-176^o

Figure 4: We apply the same technique to analyse the orientation of an opponent AIBO. The relationship between M_{S_x} (green) and M_{B_x} (yellow) reveals the robot's orientation.

very large compared to $Ratio_A$ because the SIDE of the object is closer to the viewpoint than the FRONT.

The $Ratio_B$ compared to $Ratio_A$ affect the computation of M_{B_x} and M_{S_x} in the following ways. M_{B_x} will be influenced by all of the extra pixels now visible in the SIDE of the object and thus will be biased toward that SIDE of the object. M_{S_x} will be relatively unaffected by the extra pixels (as the skeleton uses only the *local* maxima of the medial axis) however the SIDE will look larger and the skeleton obtained will reflect this. Therefore M_{S_x} will also be biased toward the visible SIDE of the object. Notice in Figure 3 that, although θ remains constant, M_{B_x} and M_{S_x} move significantly toward the visible SIDE of the object in Posture B and this is reflected in their relationship with M_{R_x}. Thus it is enough to compute $\mid M_{R_x} - M_{B_x} \mid$ or $\mid M_{R_x} - M_{S_x} \mid$ to determine whether our relative orientation lies in either the range $-90^o < \theta < 90^o$ or in either of the ranges $90^o < \theta \leq 180^o$ and $-90^o > \theta \geq -180^o$.

3 Application to Robot Soccer

We have been able to successfully detect the orientation of an opposing robot using our technique, regardless of the distance it is from the camera. The AIBO robot, in RoboCup uniform, has coloured patches on its body that are either blue or red (signifying the team) [1]. The patches are symmetrical along the long axis of the dog but are not symmetrical along any other axis. (Refer to Figure 2.1.) Once the image has been segmented, the first task is to determine which blobs of the correct colour belong to a particular dog. We do this by proximity clustering [8]. Although somewhat in exact, it is usually possible for us to identify which blobs (representing patches on uniforms) belong to which AIBO.

Once all the blobs belonging to each individual AIBO have been identified, the orientation of the AIBO is identified by symmetry using the technique above. Refer to

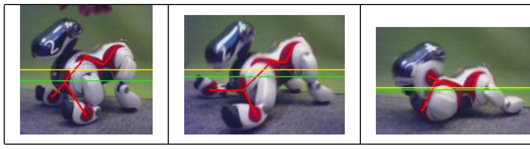


Figure 5: The relationship between M_{S_y} (green) and M_{B_y} (yellow) can be used to tell when the AIBO is performing a kick.

Figure 2.1. The medial axis each of the blobs is computed and skeleton obtained for the AIBO. The median point of this skeleton, M_{S_x}, is compared to the median point of all of the pixels in each of the coloured blobs, M_{B_x}. If the AIBO is in an orientation that exposes only its exact FRONT, then these two median points will closely match. However, due to the non-symmetrical nature of the other patches on the AIBO, M_{S_x} will be biased further from M_{B_x} and toward the SIDE of the AIBO in view as the perceived orientation angle becomes greater. Using this technique we can determine the orientation of the robot without any shape or posture analysis. Our technique is also independent of the distance to the AIBO because the constant max_d may be normalized with respect the size of the blob cluster that represents the uniform patches. Furthermore, it is robust to small changes in the posture of the AIBO. In particular we are still able to analyse an opponent while it is walking, introducing only minor inaccuracy.

We have made available on our website[3] several videos of an AIBO using our technique to recognise the posture of another AIBO. In each video the first AIBO must determine the other AIBO's posture in relation to it, and move into a position to mark it. When it reaches its marking position it assumes a guard posture to indicate that it is finished. The videos show that the AIBO maintains the ability to recognise the posture of the opposing AIBO as it moves around the perimeter, and as it moves further and closer to the opponent - only occasionally making the wrong decision. We illustrated our ability in the technical challenge section of the RoboCup 2004 competition in Lisbon, Portugal.

We have used a similar technique to analyse the intent of opposing robots in the soccer game. As an AIBO kicks the ball its vertical symmetry changes. Refer to Figure 3. From this sequence of images we see that the M_{S_y} is significantly biased *toward* M_{B_y} as the AIBO flattens its vertical posture. Since most kicks involve this flattening process we have an orientation independent way of determining if the opposing AIBO is currently kicking the ball. We have also made available a video of the AIBO using this technique to correctly predict kicks from an opponent and intercept them. We have

[3] http://www.griffith.edu.au/mipal/

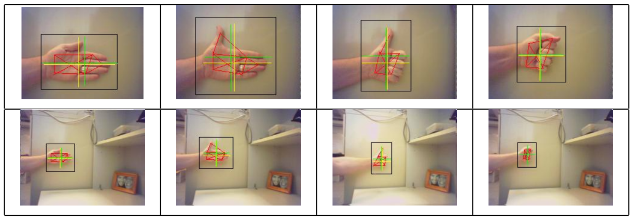
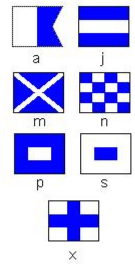
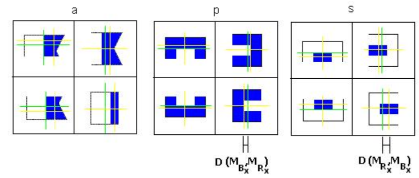


Figure 6: Symmetry of the hand may be used to recognise simple hand gestures. The thumb and fingers, when extended, create an asymmetry which is easily recognised by our technique.



(a) The subset of blue and white signaling flags.



(b) Repeated application in different regions of each flag can be used to analyse flags that would have identical symmetry otherwise.

Figure 7: Maritime signaling flag analysis.

demonstrated our ability both with and without a ball present to show that we are not using the location of the ball, but rather the features of the opponent AIBO, to determine the interception.

4 Application to Gesture Recognition of a Hand

We now illustrate how our method can be applied to differentiating between several common and simple hand gestures. Computing both the horizontal (M_{S_x}, M_{B_x}) and vertical (M_{S_y}, M_{B_y}) median values leads to a robust algorithm that is able to recognise several hand gestures. Refer to Figure 4 where we see that it is possible to accurately determine the difference between an open hand and a closed hand with the thumb either up or down. Again, M_{B_x} and M_{B_y} are illustrated by the yellow horizontal and vertical lines, while M_{S_x} and M_{S_y} are illustrated by green lines.

We see that in the images where the thumb is up, the horizontal line of (near) symmetry is broken and so M_{S_x} moves away from M_{B_x}. Similarly when the hand is extended, the vertical line of (near) symmetry is broken and so M_{S_y} moves away from M_{B_y}. Using this technique we have given the AIBO the capability to recognise and respond to four simple hand gestures. Notice that it does not matter how far away the hand is from the camera, as long as it is close enough that the details can be clearly seen. This is also illustrated in Figure 4.

5 Application to Maritime Signal Flags

The final illustration of our technique applies to maritime signaling flags. This example shows how repeated application of our technique constructs a very efficient tree to discriminate between several similar objects.

Although ships communicate more frequently with radio and modern communication devices, the maritime signaling flags have not become entirely redundant. Vessels still use combinations of flags to communicate with each other, particularly in crowded areas such as ports and channels. In the internationally agreed upon protocol there is one flag per letter of the alphabet. Signalers may either chose to spell out messages, or alternatively, many common messages are communicated by the presence of only one flag, or the combination of two. We have successfully managed to apply our technique to quickly and accurately recognise each of the alphabetical flags. Firstly, the colours in the flag are identified using image segmentation and the flags are rotated in the image so that the horizontal axis of the flag is parallel with the bottom of the image. It is not necessary to compute the skeleton for such a simple, planar recognition task so instead of using M_S we compute the medial pixel of the blue blob (M_B) and compare it to the medial point of the entire flag (M_R).

All the flags containing only blue and white are shown in Figure 7 (a). These are A, J, M, N, P, S and X. We have chosen to illustrate our technique on this subset because there are several flags in it that have similar symmetries: the symmetries are identical in, say, S and P (a white box in a blue square and a blue box in a white square)[4].

Repeated application in different regions of the bounding box allows us to analyse flags that have identical symmetries. We first take the bottom and top two thirds, then left and right two thirds. Notice that the flags which originally contained the same symmetry information become quite different. For example, in Figure 7 (b) we see that the white square inside the blue box (S) is easily distinguished from the blue square inside the white box (P) because the distance from M_{B_x} (shown in yellow) to M_{R_x} (shown in green) is different in each case. By applying this technique we can quickly and correctly identify every flag in the set.

[4] The entire set of flags can be found at http://www.omniglot.com/writing/imsf.htm.

6 Conclusion

Gesture and posture recognition are generally very computationally expensive processes. Due to the nature of the problem, they often involve very domain specific knowledge in order to correctly determine features and their relation to each other.

We have presented a useful, computationally inexpensive algorithm that works by examining the symmetry of the medial axis. Furthermore we have shown it to be a feasible heuristic by using it across a variety of domains. We have demonstrated the versatility of our technique with examples from three very different domains. We have applied our technique to the RoboCup Four-Legged League soccer competition where we have enabled one AIBO to correctly judge the posture and intention to kick of another AIBO. We have also used our technique to recognise a set of simple hand gestures and also to correctly identify each flag from the set of maritime signaling flags.

References

[1] *Proceedings of the Eighth International Robocup Symposium*, Lisbon, Portugal, 2004. Springer.

[2] C. Bajaj. Proving geometric algorithm non-solvability: An application of factoring polynomials. *Journal of Symbolic Computation*, 2:99–102, 1986.

[3] F. Chin, J. Snoeyink, and C.Wang. Finding the medial axis of a simple polygon in linear time. In *Lecture Notes Computer Science, 1004*, pages 382–391. Springer, 1995.

[4] K. Cho and S. Dunn. Learning shape classes. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 882–888. IEEE, 1994.

[5] R. Duda, P. Hart, and D. Stork. *Pattern Classification.* Wiley, N.Y., U.S.A., 2nd edition, 2001.

[6] R. Duda and P. Heart. *Pattern Classification and Scene Analysis.* Wiley, N.Y., U.S.A., 1973.

[7] V. Estivill-Castro and N. Lovell. Improved object recognition - the robocup 4-legged league. In *Intelligent Data Engineering and Automated Learning*, pages 1123–1130. Springer, 2003.

[8] J. Hartigan. *Clustering Algorithms.* Wiley, N.Y., U.S.A., 1975.

[9] G. Heidemann and H. Ritter. A neural 3-D object recognition architecture using optimized gabor filters. In *International Conference on Pattern Recognition Volume IV-Volume 7472*, pages 70–74. IEEE, 1996.

[10] N. Lovell and J. Fenwick. Linear time construction of vectorial object boundaries. In *6th IASTED International Conference on Signal and Image Processing*, pages 240–265. ACTA Press, 2004.

[11] R. L. Ogniewicz and O. Kübler. Hierarchic Voronoi skeletons. *Pattern Recognition*, 28(3):343–359, 1995.

[12] B. Shepherd. An appraisal of a decision tree approach to image classification. In *Eight International Joint Conference on Artificial Intelligence*, pages 496–501. Morgan Kaufman, 1983.

[13] R.R. Wilcox. *Introduction to Robust Estimation and Hypothesis Testing.* Academic Press, San Diego, CA, 1997.

[14] D. Wilking and T. Rofer. Realtime object recognition using decision tree learning. In *Robocup 2004 Symposium Proceedings*, pages 556–563. Springer, 2004.