Bax’s Augmented Reality Vision System

B.T. Smith and Dr. R.G. Gosine
C-CORE and Faculty of Engineering and Applied Science
Memorial University of Newfoundland
St. John’s, Newfoundland A1B 3X5, Canada
{baxter, rgosine}@engr.mun.ca
June 25, 2002

Abstract

BARViS, Bax’s Augmented Reality Vision System, is an augmented reality system utilizing image-based object recognition via support vector machines in a decision graph style classification system. The system implements a simple general architecture for augmented reality. A remote database stores only the support vectors for each object reducing the amount of data needed to be stored and transferred. The remote database is accessible to many users simultaneously, thus enabling a many-user system. The vision system is image-based and therefore does no feature selection and learns the objects in full 32x32 dimensional image space. The system is demonstrated using several objects. Results suggest that this system would be extendable to a multi-user one.
1 Introduction

We have been looking at using support vector machines at an object recognition device for an augmented reality application. Instead of computing features of an object, the support vector machines classifier uses actual images of the object itself. This image-based recognition technique has advantages over feature-based recognition techniques, since in many cases appropriate features are hard to determine.

This paper will detail the construction of BARViS - Bax’s Augmented Reality Vision System. BARViS is a wearable computer system with a head-mounted display, whereby the user could look at an object, press a button and have information about that object appear on the viewscreen. This has many applications, in particular, a tourist type environment [2], where a tourist could wear the system, and get information about buildings, restaurants, monuments and road signs just by looking at them. Getting lost would be a thing of the past since the user could at anytime get directions to anywhere. Also, if the object is not on the remote database, then the tourist could add it, so that others can use that information at a later date. BARViS uses a combination of image processing, pattern recognition, and image overlay techniques to accomplish its goal.

This paper is organized as follows. The development of BARViS is discussed in Section 2, where Augmented Reality is introduced, and the BARViS design is outlined. Section 3 gives a summary of the theory of support vec-
tor machines. Section 4 gives some results of testing BARViS. This paper is concluded with a discussion of some of the problems encountered and the future directions of BARViS.

2 Augmented Reality

Augmented Reality (AR) is the process whereby virtual information is overlayed on the physical world[1]. Many people have researched the idea [15,14] of developing such a system, and it has the potential to really impact the way we live our lives. Imagine a system that could insert critical data into your vision, in real time? Imagine mechanics who could have all the information about your car while they are looking at it. You could even do the repairs yourself! Imagine a soldier who knows all the information about any area that he/she is in, where mines are, where munitions buildings are, where hospitals are. These are just some of the possibilities that could be available.

BARViS was designed to be a multi-user augmented reality system. With this in mind, the underlying system architecture can be divided into three subsystems. It consists of an Augmented Reality Sub-System, a Database of Objects and a Location Identifier. See Figure 1. The idea is that each end-user will implement the Augmented Reality Sub-System only. The Database of Objects Sub-System resides on a computer connected to the internet and can be accessed through tcp/ip. A wireless connection to the internet provides access to the Database of Objects Sub-System. The Location Identifier
Figure 1: BARViS High-Level Architecture.

Sub-System is simply a GPS receiver. Each of these sub-systems will be discussed next.

2.1 Augmented Reality Sub-System

The Augmented Reality System Block of Figure 1 can be reduced to several other components. These include: the camera, image processing, object classification, augmented reality overlay and virtual reality display system blocks. See Figure 2.

The algorithm flow is as follows. A 640x480 color image is grabbed from the camera and is fed to the image processing and augmented reality overlay systems. The image processing system extracts a section of the image and formats it to be input to the object classification system. The object classification system first gets location information from the Location Identifier System, then downloads the classification data for that location from the Database of Objects. This image is then run through the classifier (support vector machines) and feeds the classification result to the augmented reality
overlay system. Using this classification result, the augmented reality overlay system retrieves information about the classified object from the Database of Objects, formats it, and overlays it over the original grabbed image. This image is then displayed via the virtual reality display system.

2.1.1 Camera

The camera used in BARViS is a Pyro Firewire Webcam with a Firewire-to-PCI card. This particular camera was chosen for several reasons. First, it can grab 640x480 color images at 30 frames per second, which is comparable to more expensive camera systems. Second, a frame grabber is not needed. The camera attaches to the pc via a firewire (aka 1394) port.

2.1.2 Image Processing

A 640x480 color image is sent to the image processing system block from the camera. A 100x100 area centered at 320x240 is copied from the image. This 100x100 color image is then converted to a grayscale image and reduced to a 32x32 grayscale image. Bicubic Interpolation is used to reduce the image.
This 32x32 grayscale image is then sent to the object classification system.

All the image processing is completed using OpenCV. OpenCV is Intel’s Open Computer Vision Libraries and includes Intel’s Image Processing Libraries. OpenCV is a cross-platform library of computer vision functions. Cross-platform means that it can be used with Windows and Linux. This is a necessary consideration for future implementations of BARViS since different end users will have different operating systems. Designing with cross-platform use in mind will lead to a more universal system in the long run.

2.1.3 Object Classification

This is the most complex component of the whole system. Once, the 32x32 grayscale image is retrieved from the image processing system, classification information is loaded from a database based on the Location Identifier.

The support vector machine data loaded from the database is then used to classify the 32x32 grayscale image just received. The next section gives a detailed introduction to SVM theory. This subsection will just explain how it is used here. The classification process itself follows the one-to-one format, where the input image is compared with one from the database. The winner of this classification gets compared to the next object in the database. This comparison continues until all objects have been compared. The resulting winner gets sent to the augmented reality overlay system for further processing.
2.1.4 Augmented Reality Overlay

From the winning object sent from the object classification system, the augmented reality overlay retrieves the title and text description about that object from the database. This information is then formatted and overlaid on the original 640x480 color image retrieved from the camera. See Figure 3.

2.1.5 Virtual Reality Display

The overlaid color image is then displayed on the virtual reality display. Any display device will work, although the virtual reality goggles give the system an augmented reality "feel".

2.2 Database of Objects Sub-System

There are several reasons for using the Database of Objects Sub-System. The first is to limit objects to be used in classification. Based on a Location
Identifier, all the data can be sorted such that only the objects relevant to that specific location will be used in classification. Secondly, to store all relevant information needed to distinguish between those objects. That is, all the variables used by SVMs to classify the object in question. This is summarized below. Lastly, the name and description of the objects for overlay onto the vision system are also stored in this database.

The database used is a Microsoft Access Database. This allows many points of access to the data. For example, BARViS accesses the data through ODBC (Open Database Connectivity) drivers, but the data itself can be viewed through Microsoft Access. The ODBC drivers make the database accessing platform independent. This database can be stored locally or on a separate server on the network. The advantages to having the database stored on a separate computer is that many users can connect to it and it relieves the wearable computer from all database computations.

To summarize, the data stored in the database is as follows:

- Database Contents table to highlight the contents of this database and corresponding location identifiers.

- Support Vector Machines classification constants for each of the objects to be identified. This includes A, B, nsv, alphas, and SVM data points. A and B are 1024x1 vectors of normalization data, nsv is the number of support vectors used, alphas are a (nsv)x1 vector of lagrange multipliers to multiply each data point by and SVM data points are 1024x(nsv)
matrix of the Support Vectors chosen for a given object comparison.

- Description table which contains text (title and short description) information on each of the objects to be identified.

### 2.3 Location Identifier Sub-System

To take a given scene and try to identify objects is near impossible just by comparing objects stored in a database since the number of objects is tremendous. If the scene can be limited to a distinct number of objects then the identification of objects becomes more plausible. Most AR systems use this to their advantage. To narrow down the possible objects in a given scene, a typical AR system uses some sort of Location Identifier, usually GPS. This will tell the AR system where it is. The Database of Objects can then tell it what possible objects are around that location. For example, you won’t find the CN Tower in St. John’s.

The present BARViS architecture is a reduced version of the one shown in Figure 1. BARViS does not have access to a GPS, therefore the Location Identifier will be limited to one region only. See Figure 4. This limitation will be eliminated in future versions of BARViS.

### 3 Support Vector Machines

Support vector machines were invented by Vladimir Vapnik and his team at AT&T Bell Laboratories. Since then many improvements
and applications have been discovered \cite{8, 7, 5, 9}. Support vector machines seem to be well suited to image pattern recognition \cite{5, 13}. Although they are a bi-class classifier, methods have been devised to extend support vector machines to multi-class classification \cite{11} including one-to-all and one-to-one approaches.

To determine what object the user is looking at, BARViS uses support vector machines configured in a top-down decision graph type of multiclass classifier based on the combination of biclass SVMs \cite{11}. This is illustrated in Figure 5 for the case of four classes. To classify a given object, the computation of all the possible biclass SVM classifiers is required, each trained on a pair of classes. All classes must then be compared to each other and the ultimate winner is the classification result.

Essentially, a SVM finds the hyperplane $w \cdot x + b$ which separates two classes with the most generality. The hyperplane is composed of $w$ the weight vector, $x$ the vector of features, and $b$ the bias term. This best hyperplane is the one which maximizes the distance or margin between the two classes. This can be extended to non-linear domain as well.
Figure 5: SVM Multiclass classification technique.

Figure 6: Application of OSH to two datasets.

All of the nodes (except the end nodes) in the decision graph in Figure 5 represents a biclass SVM and has two children. When a vector is input into the graph, it starts at the root node and follows the decision path along the graph until it reaches an end node. Then the classification is complete.

3.1 Optimal Separating Hyperplane

Given two classes of data to be separated, there are a number of lines that can separate them, but there is only one that maximizes the distance between it and the nearest data point of each class, or margin. This line gives the 'best' results, where 'best' means that it gives the highest classification rate when new data is used. This line should generalize well compared with other ones. It is called the Optimal Separating Hyperplane (OSH). Figure 6 shows the application of the OSH.
The OSH algorithm is based on finding two parallel lines which separate the data and maximize the perpendicular distance between them. The idea is that a third line parallel to and between these two lines provides a good approximation to the 'best' separating hyperplane. So, once the OSH is found from the training data, as a mathematical function, then simple geometry can be used to calculate which side of the line a new data point will lie on and thus its classification. SVMs use geometric properties to calculate the OSH directly from the training data.

Given the following training data:

\[(x_1, y_1), \ldots, (x_m, y_m) \mid x = \text{real}, y = \{+1, -1\}\]  

(1)

where each data point is described by a feature vector \(x_i\) and a classification value \(y_i\). Note that \(y_i\) can have values of +1 or -1 depending on its class. Hyperplane one is required to pass through at least one data point of class one. The same is true for hyperplane two and class two. There can be no points between the two hyperplanes. The OSH is another hyperplane, parallel to and half way between the first two hyperplanes. This hyperplane defines the boundary between the two classes. The data points that the outer hyperplanes pass through are called Support Vectors. The first outer hyperplane is described by,

\[(w \cdot x) + b = +1\]

(2)
where it goes through a data point of class \( y = +1 \). The second outer hyperplane is described by,

\[
(w \cdot x) + b = -1 \tag{3}
\]

where it goes through a data point of class \( y = -1 \). The constants \( w \) and \( b \) define the hyperplanes, with \( w \) being perpendicular to the hyperplanes and \(-b/\|w\|\) being the perpendicular distance from the origin to the middle hyperplane. The right-hand side of Equation 2 will be greater than or equal +1 for all points of class \( y = +1 \). The right-hand side of Equation 3 will be less than or equal -1 for all points of class \( y = -1 \). These two equations can combined into one equation describing a constraint on all the data points,

\[
y_i[(w \cdot x_i) + b] \geq 1, \forall i \tag{4}
\]

The perpendicular distance between the two outer hyperplanes or \textit{margin} is equal to \( 2/\|w\| \). Therefore, finding the hyperplanes with the largest margin reduces to finding values for \( w \) and \( b \) that maximize \( 2/\|w\| \) or equivalently minimize \( \frac{1}{2}(w \cdot w) \), subject to the constraint in Equation 4.

A standard method for handling optimization problems with constraints is through the minimization of the Lagrangian. To take the constraints into account, the addition of terms involving Lagrange multipliers to the objective
function is necessary. This results in the following Lagrangian,

\[ L_p = \frac{1}{2} \| w \|^2 - \sum_{i=1}^{m} \alpha_i y_i (w \cdot x_i) + b + \sum_{i=1}^{m} \alpha_i \]  

(5)

where \( \alpha_i \) are the Lagrange multipliers associated with each of the constraints in Equation 4. The Lagrangian has been minimized with respect to the primal variables \( w \) and \( b \), and maximized with respect to the dual variables \( \alpha_i \). This means that a saddle point exists. At the saddle point, the derivatives of \( L_p \) with respect to the primal variables must be equal to zero. This yields,

\[ w = \sum_i \alpha_i y_i x_i \]  

(6)

and

\[ \sum_i \alpha_i y_i = 0 \]  

(7)

and from the definition of Lagrange multipliers, we get,

\[ \alpha_i \cdot (y_i (w \cdot x_i + b) - 1) = 0, \ i = 1..m \]  

(8)

Inserting Equations 6 and 7 into 5 removes the primal variables and results in the Wolfe dual Lagrangian where we just have to find the \( \alpha_i \) which maximize:

\[ L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \]  

(9)

subject to \( \alpha_i \geq 0 \), for all \( i \), and Equation 7.
This works out well because the problem is now reduced to finding the Lagrange multipliers (the dual variables) that maximize Equation 9 and satisfy both the non-negative constraints and the constraints of Equation 7. Equation 8 means that only those data points which lie on the outer hyperplanes will have non-zero Lagrange multipliers. These data points are called the support vectors and are the points that determine the position of the hyperplanes. The other points do not affect the solution at all, and so can be removed entirely.

Equation 9 can be solved using any quadratic programming solver. Once the Lagrange multipliers are known, the solution for w is given by Equation 6, where the sum is over the support vectors, since they are the only ones with non-zero $\alpha$. Equation 8 yields b using any of the support vectors, although one generally averages over all the support vectors for better accuracy. Once w and b are known, the classification of an unknown data point, v, is given by the sign of,

$$ b + \sum_i \alpha_i y_i x_i \cdot v $$

(10)

where the sum is over the support vectors. This determines on which side of the OSH that the data point lies.

### 3.2 Extending OSH to Nonlinearly Separable Classes

The idea of OSH can be extended to distinguish between nonlinearly separable classes. The input space (ie the 32x32 pixel image) is mapped into
Figure 7: Application of a non-linear separating hyperplane to separate the data.

a high-dimensional feature space through some non-linear mapping function and then the OSH is constructed in this feature space. This linear decision surface in feature space corresponds to a non-linear decision surface in input space. In other words, an input vector, $x$, gets mapped into a high dimensional feature space, $z$, through a non-linear transformation, $\Phi$. The most common mappings are polynomials, radial basis functions and various sigmoidal functions.

If another point is added the the set of points in the previous example, an OSH can’t be used to separate the data. See Figure 7. Using a non-linear separating hyperplane, however, which is equivalent to mapping into a high dimensional space, a separation is possible.

To implement this mapping the Lagrangian in Equation 9 gets transformed to:

$$L_D = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \Phi(x_i) \cdot \Phi(x_j) \quad (11)$$
and the classification relation in Equation 10 becomes:

\[ b + \sum_{i} \alpha_i y_i \Phi(x_i) \cdot \Phi(v) \]  \hspace{1cm} (12)

Since Equations 11 and 12 depend only on the dot product between the two transformed feature vectors, a kernel function can be used:

\[ K(x, y) = \Phi(x) \cdot \Phi(v) \]  \hspace{1cm} (13)

and the transform, \( \Phi \) doesn’t have to get computed explicitly. Equation 12 then becomes:

\[ b + \sum_{i} \alpha_i y_i K(x_i, v) \]  \hspace{1cm} (14)

with the test feature vector now inside the summation of the support vectors.

In general, the mapping \( \Phi \) will be to a higher dimensional space. Since the solution is still to a linear problem, just in a different space, the computational overhead is essentially the same. The solution and parameters for the hyperplane are in the higher dimensional space and when one transforms back to the original space the boundary becomes non-linear.

4 Results

Random objects around the Intelligent Systems Laboratory (ISLAB) were chosen to test the system. These objects include a remote control truck, a
Figure 8: 32x32 intensity images of each object used for training the support vector machines classifier.

telephone, a fan and a wall mounted network hub. See Figure 8. Twenty-five pictures of each object were obtained at different angles and used to train the SVM classifier. Training of the SVM occurs offline and uses Osuna’s implementation of SVMs [8]. The number of support vectors used in each bi-class comparison is shown in Table 1.

<table>
<thead>
<tr>
<th>Objects being compared</th>
<th>Number of Support Vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>RC Truck and Telephone</td>
<td>19</td>
</tr>
<tr>
<td>RC Truck and Fan</td>
<td>21</td>
</tr>
<tr>
<td>RC Truck and Network Hub</td>
<td>22</td>
</tr>
<tr>
<td>Telephone and Fan</td>
<td>16</td>
</tr>
<tr>
<td>Telephone and Network Hub</td>
<td>14</td>
</tr>
<tr>
<td>Fan and Network Hub</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 1: Number of support vectors per bi-class comparison

Results are very promising. The classifier responds well to translations and scaling of the object to be classified. Even when the image is very offcenter the system still manages to identify it correctly. Consider the object in Figure 9. It’s displacement from the center of the identification window is almost half the object width and the system still identifies it correctly.
Similar results are obtained when scaling the image. Figure 9 shows that the object is correctly classified over almost a doubling of object scale.

5 Conclusions and Future Work

This paper has introduced BARViS, Bax’s Augmented Reality Vision System, designed to be a multi-user visual information system. Using image-based recognition with support vector machines allows for feature independent classification.

Using BARViS, the user can identify objects in a scene, based on information stored in the Database of Objects and the Location Identifier. In future releases the user will be able to store objects to the Database of Objects for others to use in the future. GPS will also be added in future versions.

References


[13] Roobaert D 1999 Improving the generalization of linear support vector machines: an application to 3D Object Recognition with Cluttered Background. *Proceeding SVM workshop at the 16th International Joint Conference on Artificial Intelligence (IJCAI99)*
