Support Vector Machines for Cinematography Real-Time Camera Control in Storytelling Environments

Edirlei E. S. de Lima[1] Cesar T. Pozzer[1] Eduardo C. D. Favera[1] Marcos C. d'Ornellas[1]
Angelo E. M. Ciarlini[2] Bruno Feijó[3] Antonio L. Furtado[3]

[1]UFSM, Depto. de Eletrônica e Computação, Brasil
[2]UNIRIO, Depto. de Informática Aplicada, Brasil
[3]PUC-Rio, Depto. de Informática, Brasil

Figure 1: Camera shots selected by the director using support vector machines.

Abstract

This paper proposes an intelligent cinematography director for camera control in plot-based storytelling systems. The role of the director is to select in real-time the camera shots that best fit for the scenes and present the content in an interesting and coherent manner. Director's knowledge is represented with a collection of support vector machines (SVM) trained to solve cinematography problems of shot selection. With this work we introduce the use of support vector machines, applied as an artificial intelligence method, in a storytelling director. This approach also can be extended and applied in games and other digital entertainment applications.

Keywords: Storytelling, Cinematography, Artificial Intelligence, Support Vector Machine.

Authors' contact:
edirlei@msn.com
{pozzer,ornellas,favera}@inf.ufsm.br
angelo.ciarlini@uniriotec.br
{bfeijo,furtado}@inf.puc-rio.br

1. Introduction

Current advances in graphic technologies are paving the way to realistic digital entertainment applications. However, with this evolution new challenges have emerged. One area that deserves emphasis and has been the target of several researches in last the years is the application of cinematography in games and storytelling applications.

Cinematography is defined as the art of film-making. It consists of techniques and principles that control how a film should be produced and filmed. Most of the principles of cinematography are about how a camera should be used in order to accomplish tasks such as engaging the interest of the viewer, enhancing and clarifying the narrative, and presenting the content in an interesting and coherent manner. Viewers are used to a general storytelling pattern. Therefore, when they watch a single movie, they unconditionally try to impose a pattern of his/her own.

In this paper, we focus on the application of cinematography concepts to storytelling applications. Interactive storytelling is a new medium of digital entertainment where authors, audience, and virtual agents engage in a collaborative experience. It can be seen as a convergence of games and filmmaking. Storytelling systems can be divided in two different models. The first model corresponds to the character-based approach [Cavazza et al. 2002; Mateas and Stern 2000; Young 2000] where the storyline usually results from the real-time interaction among virtual autonomous agents that usually incorporates a deliberative behavior. The main advantage of a character-based model is the ability of anytime user intervention. As a result of such strong intervention, there is no way to estimate what decisions or actions will be made by the virtual actors. The director does not have then the same control over the process as it usually occurs in real filmmaking. The other model corresponds to the plot-based approach [Grasbon and Braun 2001; Spierling et al. 2002], where characters incorporate a reactive behavior, which follows rigid rules specified by a plot. The plot is usually built in a stage that comes before dramatization. This approach ensures that actors can follow a predefined script of actions that are known beforehand. The script may be

built automatically from a plot or with the help of the author.

To apply cinematography concepts in storytelling applications there are two most common approaches. The first approach is the use of film idioms, which represents the most usual way to present a specific type of scene. Idioms are used in works such as Charles et al. [2002]. The second approach is the division of the system in different modules or agents that represent the various roles people play in a movie set, such as in Hawkins [2004]. In other works, such as Courty et al. [2003] both approaches are used. However, these works have only superficially incorporated cinematography rules.

This paper proposes a cinematography director for plot-based storytelling systems. The director uses a collection of Support Vector Machines (SVM) trained with cinematography knowledge to select, in real-time, the best shots for the dramatization of scenes. The rest of the paper is organized as follows: section 2 compares our approach with previous research. Section 3 presents the principles and concepts of cinematography. Section 4 presents our system architecture. Section 5 brings a detailed look at the director implementation. In section 6, we analyze the performance and accuracy results to demonstrate the efficiency of our approach. Finally, in section 7 we present the concluding remarks.

2. Related Works

Many works have already been done with the objective of applying concepts of cinematography in games. The basic principle of camera positioning employing cinematography knowledge in form of idioms was first explored by Christianson et al. [1996]. These idioms encapsulate the combined knowledge of several personal roles in a traditional filming set and are widely used in research involving camera systems. However, film idioms are only able to solve the problem of direct manipulation of the virtual camera. In other works, such as Hawkins [2004], the system is divided in different modules or agents, the most common approach consider three elements: director, editor and cinematographer. This approach is known to be a better solution since some cinematography rules do not only involve the camera manipulation.

In research involving cinematography applied to storytelling systems, there is a clear distinction between the techniques that can be applied to character-based and plot-based approaches. Plot-based applications give access to all the actions before camera planning, allowing the system to have a greater control of the scenes based upon pure cinematography knowledge. Character-based applications do not allow the same level of control over the scenes, making camera planning more complicated, since all information is sent in real-time to the camera system.

The first camera system in character-based storytelling applications was developed by He et al. [1996]. They organized film idioms as nodes of hierarchical trees. Each idiom operates as a state machine and defines the scene shots to be used. Halper et al. [2001] has proposed a camera control based upon constraint specifications; however high constraint satisfaction implies in poor frame coherence. Charles et al. [2002] have explored architectural and organizational concepts to achieve satisfactory camera planning when we have different context timeline stories that can be alternated with the flow of the time. In plot-based applications, Courty et al. [2003] introduces a scheme for integrating storytelling and camera systems.

Current approaches only reach superficial implementation, and do not provide a good dramatization quality to become comparable with a real movie. In this work, we try to contribute towards this goal by proposing a novel approach for the architecture and implementation of a cinematography virtual director.

3. Principles of Cinematography

The term cinematography was created in the film industry a long time ago to describe the process of creating images on film. With the advancement of industry and the emergence of new technologies in digital video with high definition formats, the tern are expanded. Now it is understood as a generic term covering all aspects of camera work, including the creative aspects involved with making aesthetically pleasing images and the technical aspects involved with using cameras, lights, and other equipment [Newman 2008].

Although a film can be considered a linear sequence of frames, it is often helpful to think of a film as having a structure. At the highest level a film is a sequence of scenes. Each scene is composed of a number of shots, a shot being a continuous view filmed by one camera without interruption. The transition from one shot to the next is known as a cut.

The size of the image on the film is determined by the distance of the camera from the subject. The closer is the camera, the larger is the image. This distance defines a shot type. Supposing that the subject is a character, an example of shot type is the medium shot, which depicts characters from the thighs to above the head. Another example is the close-up, which depicts them from the chest to above the head [Mascelli 1998].

The type of camera angle strongly influences the way a scene is perceived by the viewers. It also defines how viewers may become part of the action. When a choice is made to the objective angle, the viewer sees

the event on screen as if an unseen observer [Mascelli 1998]. A subjective camera angle makes the viewer a part of the scene.

Another important aspect of filmmaking corresponds to the camera movements. They affect the aesthetic and psychological properties of a scene. An example of camera movement is the tracking, when the camera moves alongside a character while filming, giving to the viewers the feeling that they are walking alongside the character [Mascelli 1998]. Movements should be executed in such a way that the viewer does not get disoriented.

Cinematography is a complex process, and many rules demand human interpretation of the scenes to be correctly applied. However, cinematographers have defined some heuristics for selecting good shots [Arijon 1976]. Some examples are:

- **Create a line of action:** this line should connect the two major points in one scene (most of the times, the two actors that interact in the scene);
- **Parallel editing:** Scenes should alternate between different contexts, locations and times;
- **Show only peak moments of the story:** Repetitive movements should be eliminated;
- **Don't cross the line:** Once a scene is taken by a side of the interest line, the camera should in principle keep itself in that side, not making unexpected movement shots. The camera can switch sides, but only upon an establishing shot, that shows that transition;
- **Let the actor lead:** The actor should initiate all movement, and the camera should come to rest a little before the actor;
- **Break movement:** A scene illustrating a movement must be broken in two shots at least.

4. System Architecture

Our storytelling system architecture is organized in four modules. The Scriptwriter is responsible for controlling the plot and the story flow; the Scenographer is responsible for creating and arranging the sceneries; the Director defines how scenes will be filmed; and the Cameraman is responsible for positioning the cameras. Figure 2 shows a diagram of the architecture.

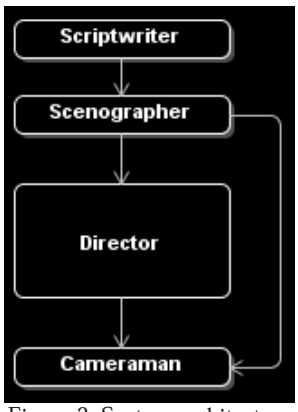


Figure 2: System architecture

The main component of our architecture and the focus of this work is the Director. It concentrates the cinematography knowledge and decides, in real-time, the best way to present scenes. The knowledge is represented by means of several support vector machines trained to solve cinematography problems involving camera shot selection. Support vector machines are used as an effective method for general purpose pattern recognition; they are based on statistical learning theory and are specialized for small sample sets [Tyagi 2008]. A similar approach is used by Passos et al. [2008] to select camera shots in a race car game using a neural network classifier. Support vector machines have better generalization than neural networks and guarantee local and global optimal solutions similar to those obtained by neural networks [Gunn 1998]. In recent years, support vector machines have been found to be remarkably effective in many real-world applications such as in systems for detecting microcalcifications in medical images [El-Naqa et al. 2002], automatic hierarchical document categorization [Cai et al. 2004], spam categorization [Drucker 1999], among others.

In our system, the modules are agents that communicate with each other by means of message exchange and can be summarized as follows:

1. The Scriptwriter reads the information about the current scene from the story plot and sends it to the Scenographer;
2. The Scenographer prepares the actors and scenario for the scene dramatization and also places objects and involved actors in the scene. The information about the scenario is sent to both the Cameraman and the Director;
3. The Cameraman, following cinematography rules, places a set of cameras in the scene for all possible shots for the current scene;
4. The Director extracts from the scene all important data and applies them to a support vector machine to select the best shot for the scene. This information is then sent to the Cameraman;

5. The Cameraman activates the shot selected by the Director and, if necessary, executes a camera movement or zooming operation.

5. The Director

In a film production, the director creatively translates the written word into specific images. He visualizes the script by giving to abstract concepts a concrete form. The director establishes a point of view on the action that helps to determine the selection of shots, camera placements and movements. The director is responsible for the dramatic structure and directional flow of the film.

In our system, the role of the director is to choose which shot should be used at each time to highlight the scene emotion and to present the content in an interesting and coherent manner. To perform this task, the director uses a collection of support vector machines trained to classify the best shots for the dramatization scenes.

The process consists of two steps. First, the training process, which is done before the story dramatization, consists in simulating some common scenes and defining the solution for the shot selection. The features of these scenes, actors and environment are used to teach the support vector machine how to proceed in this situation in order to detect similar situations in the future. The second step is the prediction process that is done in real-time during the dramatization by using the knowledge acquired through the training process to predict (classify) an unknown situation. Subsequent sections detail all this process.

The input of our support vector machines are the important features from the environment, scene, and involved actors. The output is the selected shot that best matches with the input features, as shown in Figure 3.

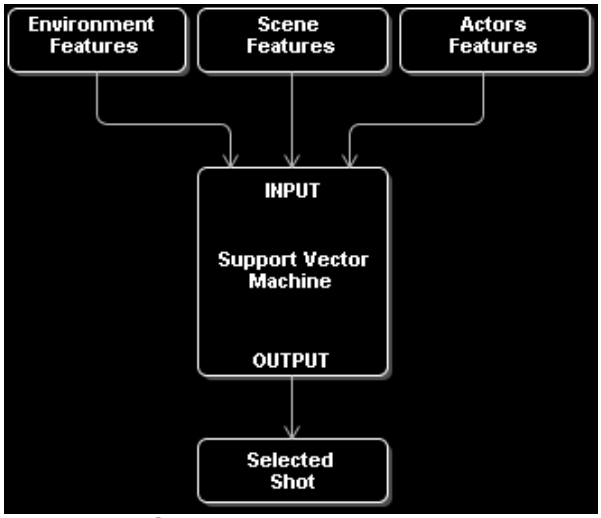


Figure 3: Support vector machine input and output

5.1 Support Vector Machine

The support vector machine, proposed by Vapnik [1995], is a powerful methodology for solving machine-learning problems. It consists of a supervised learning method that tries to find the biggest margin to separate different classes of data. Kernel functions are employed to efficiently map input data, which may not be linearly separable, to a high dimensional feature space where linear methods can then be applied.

The original idea of SVM is to use a linear separating hyperplane to separate the training data set into two classes. Figure 4 shows an optimal hyperplane separating the blue class from the green class.

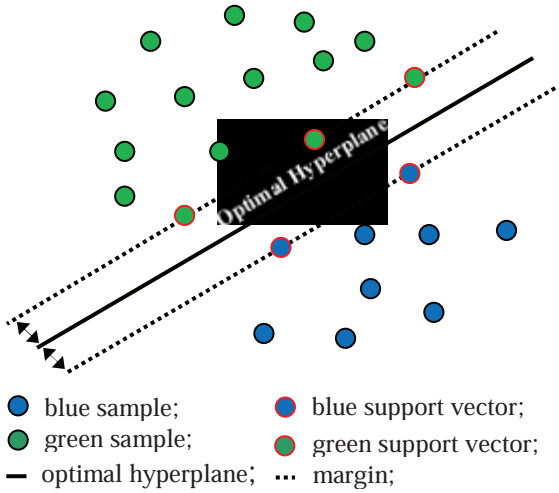


Figure 4: Optimal hyperplane separating two classes

Suppose the training data $(x_1, y_1), \cdots, (x_l, y_l)$, where each sample $x_i \in R^n$ belongs to a class $y_i \in \{-1, +1\}$.

The boundary hyperplane can be as follows:

$$\omega \cdot x + b = 0$$

and the separate margins as:

$$\omega \cdot x + b = +1$$
$$\omega \cdot x + b = -1$$

where,

- ω is a weight vector;
- b is a bias;
- x is a point in the space R^n.

This set of vectors is separated by the optimal hyperplane if and only if it is separated without error and the distance between the closest vector and the hyperplane is maximal. The separating hyperplane can be described in the following form:

$$\begin{cases} \omega \cdot x_i + v \geq +1, & if \quad y_i = +1 \\ \omega \cdot x_i + v \leq -1, & if \quad y_i = -1 \end{cases}$$

or equivalently:

$$y_i(\omega \cdot x_i + b) \geq 1, \qquad i = 1, \ldots, l$$

The optimal hyperplane is the one that satisfies the conditions and minimizes the function: $\frac{1}{2}\|\omega\|^2$.

Vapnik [1995] has shown that, to perform this minimization, we must maximize the following function with respect to the variable α_i:

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

subject to $0 \leq \alpha_i$, $i = 1, \ldots, l$ and $\sum_{i=1}^{l} \alpha_i y_i$

Those $x_i s$ with $0 < \alpha_i$ are termed Support Vectors. The support vectors are located on the separating margins and are usually a small subset of the training data set, denoted by X_{SVM}.

For an unknown vector x_i, its classification corresponds to finding:

$$f(x) = sign\left(\sum_{x_i \in X_{SVM}} \alpha_i y_i (x \cdot x_i) + b \right)$$

where

$$\omega = \sum_{x_i \in X_{SVM}} \alpha_i y_i x_i$$

and the sum is over those nonzero SVs with $0 < \alpha_i$. In other words, this process corresponds to finding which side of the hyperplane the unknown vector belongs.

However, in most cases, the classification is not so simple, and often more complex structures are needed in order to make an optimal separation. For example, in Figure 5, the separation requires a curve that is more complex than a simple line.

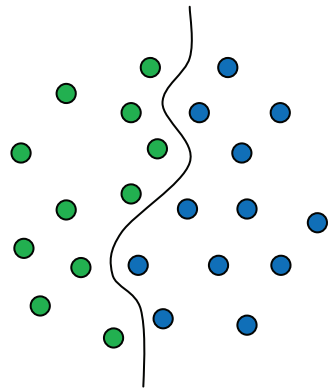


Figure 5: Non-linearly separable classes

To construct the optimal hyperplane in the case when the data is linearly non-separable, SVM uses two methods. First, it allows training errors. Second, it non-linearly transforms the original input space into a higher dimensional feature space by a function $\varphi(x)$. In this higher space, it is possible that the features may be linearly separated [Wang and Zhong 2003]. Then the problem can be described as:

$$\min \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{l} \xi_i \quad (2)$$

subject to

$$y_1(\omega \cdot \phi(x_i) + b) \geq 1 - \xi_i, \ \xi_i \geq 0, \ i = 1, \ldots, l, \ C > 0$$

A penalty term $C\sum_{i=1}^{l} \xi_i$ in the objective function takes the training errors into account. If the data are linear separable, problem (2) goes back to (1) as all ξ_i will be zero. We can equivalently maximize $W(\alpha)$ but the constraint is now $0 \leq \alpha_i \leq C$ instead of $0 \leq \alpha_i$:

$$W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j (\phi(x_i) \cdot \phi(x_j))$$

subject to

$$0 \leq \alpha_i \leq C, \ i = 1, \ldots, l \text{ and } \sum_{i=1}^{l} \alpha_i y_i = 0$$

The inner products in the high-dimensional space can be replaced by some special kernel functions. Some popular kernels are radial basis function kernel and polynomial kernel.

For example, to linearly separate the classes showed in Figure 5, the classes need to be mapped and rearranged using a kernel function in a high-dimensional space. After the mapping, classes become linearly separable and the optimal hyperplane can be created (Figure 6).

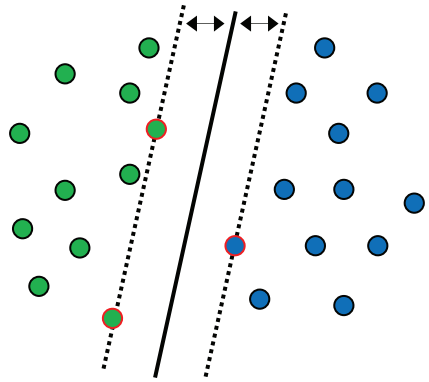


Figure 6: Classes mapped and rearranged to become linearly separable.

Support vector machines were originally created for binary pattern classification. For our problem, a multi-class pattern recognition is necessary because in most part of the scenes we have more than two possible shots for the same scene. To solve this problem, we use the "one-against-one" approach [Knerr et al. 1990] in which classifiers are constructed and each one trains data from two different classes, creating a combination of binary SVMs. The first use of this strategy on SVM was by Friedman [1996]. In classification we use a voting strategy to decide the class of the input pattern.

5.2 Training Process

Before using support vector machines to select the shots in our dramatization, they have to be trained to acquire the necessary knowledge to create the optimal hyperplane separating the shots, so that it can be used to predict the best shot for new scenes.

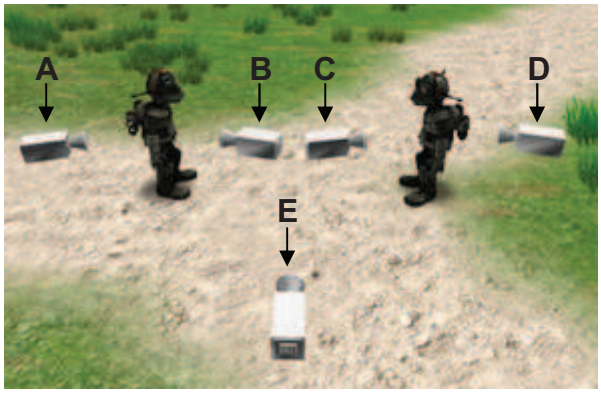
In order to train the support vector machines, we simulate some common situations that happen in real films. Based on cinematography rules and principles, we perform the selection of best shots for these scenes and store them in a database together with features from the simulated scenes. The training database is composed of several samples of simulated scenes, each one with the features and the selected shot for the simulated scene. This training database is created once and is used in all future dramatizations.

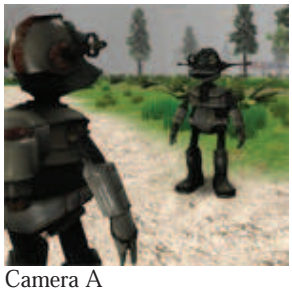
The used features are:

- Normalized values of the position (X, Y and Z) (relative to the center of scene) of the actors involved in the scene. These values influence the camera shot in action scenes when the position of the actor can change during the dramatization.
- The current emotional state of the actors involved in the scene (happiness, sadness, angriness or scariness). The emotional state in most cases influences the selected shot to highlight the emotional actor state.
- The acting or talking actor. This feature is the most important because the actor must be visible in the shot.

Numerical values are associated with the abstract types. The emotional state happiness is, for example, represented by the value 1, sadness by the value 2. All features are then normalized (between -1 and 1).

The classes are the possible shots (camera angles) for the scene. These shots are defined in our system by the Cameraman module, which, for each scene, creates a line of action and positions the cameras in an appropriated location, improving the scene visualization by following standard cinematography rules and patterns proposed by Arijon [1976]. For example, in a dialog scene between two actors (Figure 7) there are 5 possible shots (classes); camera A and camera D highlight the viewer's attention to one actor while keeping the other actor visible in the scene; camera C and camera B highlight the attention only to one actor and emphasizes his emotional state; and camera E shows both actors. For this scene, we can extract 9 features: the position X, Y and Z of the two actors (6 features), the emotional state of the two actors (2 features), and the active talking actor (1 feature).





Camera A

Camera C


Camera D


Camera B


Camera E

Figure 7: Possible camera shots for a dialog scene.

For each type of scene we have a different support vector machine; the number of features (inputs) and classes (outputs) depends on the type of scene and number of involved actors. Figure 8 illustrates this combination of support vector machines. The director has N support vector machines and each one with different inputs and outputs.

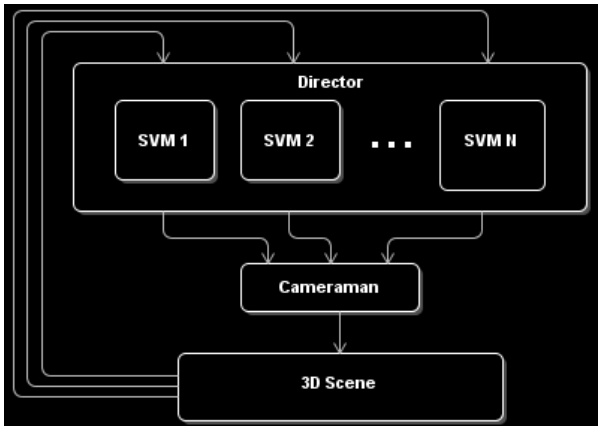


Figure 8: Director Architecture.

5.3 Predicting Process

With the support vector machines trained with cinematography knowledge, the Director module is able to act as a film director and, based on the previous experience, select in real-time the best shots to show the scenes.

To predict the best shot, the director executes the following steps:

- Selects the active support vector machine based on the type of the current scene;
- Extracts the features from the active environment and actors; these features are the same used to create the training database;
- Applies the extracted features to the support vector machine;
- Use the support vector machine output to set the active camera. The result of our support vector machine is the camera shot classified as the best solution to show the scene.

The scenes are composed by different shots; the transition between the shots occurs when an important event happens in the scene, for example when the emotional state of an actor changes or when an actor executes an action. The director detects in real-time these events and executes the predicting process to use the support vector machine knowledge to choose the new shot.

Consider a scene where the actor chases an animal (Figure 9). We have two possible shots for this scene: camera A and camera B. The director detects in real-time the type of the scene and activates the support vector machine for chasing scenes. Every time when a new support vector machine is selected an initial shot must be selected, so the director extracts from the environment the features used by the active support vector machine and apply these features to it; the support vector machine applies then the classification algorithm to determine the shot that best fits the current scene; finally, the director sends this selection to the Cameraman module which activates the selected camera. When a new important event occurs, for instance, while along the chase the actor speaks something, the director executes the prediction process again, and probably that action will influence on the selected shot.


Camera A


Camera B

Figure 9: Possible camera shots for a chasing scene

6. Results

To validate our architecture we run two tests: first the performance test, to check the necessary time to predict a new shot. The second test is the recognition rate, to check the accuracy of the predicted shots. The tests have occurred on an Intel Core 2 Quad 2.40 GHz, 4 GB of memory, using a single core to process the support vector machines.

To test the performance of our proposed solution, we trained our support vector machines with a different number of samples and use them to predict the shots for a sequence of 6 scenes, with a total of approximately 40 different shots. For each shot, we calculate the necessary time for the prediction process. Figure 10 shows performance results in a line chart with the training set size ranging from 10 to 55 samples and the times correspond to the average of the all support vector machines trained with the current number of samples.

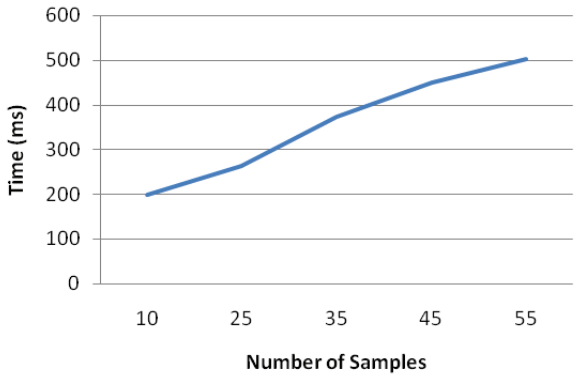


Figure 10: Prediction performance test with different training sets.

To test the recognition rate, for each support vector machine in our system, we created 5 training sets with a different number of samples and, for each one, a testing set with half the size of the corresponding training set. The training sets are used to train the support vector machine and the samples of the current test set are predicted. Correct and wrong predicted shots are then computed. Table 1 shows the computed results of this test with the training set size ranging from 10 to 55 samples. The presented percentages of accuracy correspond to the average of the results obtained for the different support vector machines.

Table 1: Recognition rate with different training sets.

Number of Samples	10	25	35	45	55
Accuracy	92%	94.6%	96.5%	98%	98.6%

Figure 11 shows the results of this test in a line chart.

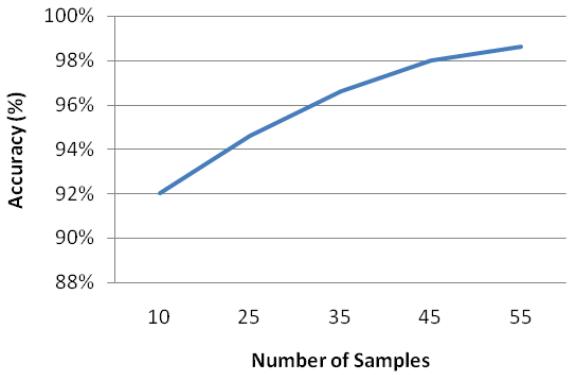


Figure 11: Recognition rate with different training sets.

It is clear that the computational cost grows almost linearly with the number of samples. More samples result in a high accuracy but in slow recognition; few samples result in a fast recognition but in a low accuracy. However, with small training sets we obtain high percentage of correct recognition of the best shots, ensuring high accuracy in the shot selection and without high computational costs.

7. Conclusion

In this paper we have presented an intelligent cinematography director that uses a collection of support vector machines trained with cinematography knowledge to select in real-time the best scene shots in storytelling dramatization. Our methodology is applicable not only to storytelling systems; it can be adapted to other entertainment applications, such as games, virtual worlds or 3D simulations.

In our tests, support vector machines showed to have excellent recognition rate (between 92% and 98%) with small training sets and without high computational cost (less that 1 second to predict). This approach ensures that most of the times the selected shots are the best solution to show the scene in accordance with cinematography principles and rules.

Support vector machine is a powerful machine learning methodology, however, still not widely explored in the area of artificial intelligence for games. In this paper, we have shown that support vector machines can be successfully applied in storytelling to select camera shots in real-time. Extending the use of support vector machines in games and entertainment computing in general is a promising approach to implement other artificial intelligence and machine learning tasks, such as controlling the behavior of non-player characters. Training our support vector machines at real-time in accordance with feedback provided by the users is also an interesting point to be investigated.

Acknowledgements

This work was supported by CAPES/RH-TV-Digital, CAPES/PROCAD, and CNPq. Authors would like to express their gratitude to Laboratório de Computação Aplicada (LaCA) - UFSM.

References

ARIJON, D., 1976. Grammar of the Film Language. Communication Arts Books, Hasting House, Publishers, New York.

CAI, T., HOFMANN, T., 2004 Hierarchical document categorization with support vector machines. In Proceedings of the ACM 13th Conference on Information and Knowledge Management.

CAVAZZA, M., CHARLES, F. AND MEAD, S., 2002. Character-based interactive storytelling. IEEE Intelligent Systems, special issue on AI in Interactive Entertainment, 17(4):17-24.

CHARLES, F., LUGRIN, J., CAVAZZA, M. AND MEAD, S., 2002. Real-time camera control for interactive storytelling. In Proceedings of the Game On, London, UK.

CHRISTIANSON, D. B., ANDERSON, S. E., HE, L., COHEN, M. F., SALESIN, D. H., WELD, D. S., 1996. Declarative Camera Control For Automatic Cinematography. In Proceedings of the AAAI '96, 148-155.

COURTY, N., LAMARCHE, F., DONIKIAN, S. AND MARCHAND, E., 2003. A cinematography system for virtual storytelling. In Proceedings of the International Conference on Virtual Storytelling, Toulouse, France.

DRUCKER, H., WU, D., VAPNIK, V., 1999. Support Vector Machines for Spam Categorization IEEE Trans. on Neural Networks , vol 10, number 5, pp. 1048-1054.

EL-NAQA, I., YANG, Y., WERNICK, M. N.; GALATSANOS, N. P.; NISHIKAWA, R. M., 2002. A support vector machine approach for detection of microcalcifications. IEEE Trans. on Medical Imaging, vol. 21, no. 12, pp. 1552-1563.

FRIEDMAN, J., 1996. Another approach to polychotomous classification. Technical report, Department of Statistics, Stanford University.

GRASBON, D. AND BRAUN, N., 2001. A morphological approach to interactive storytelling. In: Fleischmann, M.; Strauss, W., editors, In Proceedings of the CAST01, Living in Mixed Realities, Sankt Augustin, Germany, p. 337-340.

GUNN, S., 1998. Support Vector Machines for Classification and Regression. Technical Report, University of Southampton.

HALPER, N., HELBING, R., STROTHOTTE, T., 2001. A camera trade-off between constraint satisfaction and frame coherence. Eurographics, volume 20.

HAWKINS, B. 2004. Real-Time Cinematography for Games (Game Development Series). Charles River Media, Inc., Rockland, MA, USA.

HE, L., COHEN, M., AND SALESIN, D. 1996. The virtual cinematographer: A paradigm for automatic real-time camera control and directing. In Proceedings of the ACM SIGGRAPH '96, 217-224.

KNERR, S., PERSONNAZ, L., AND DREYFUS, G., 1990. Single-layer learning revisited: a stepwise procedure for building and training a neural network. In J. Fogelman (ed.), Neurocomputing: Algorithms, Architectures and Applications, Springer.

MASCELLI, J., 1998. The Five C's of Cinematography: otion Picture Filming Techniques. Silman-James Press, Los Angeles.

MATEAS, M. AND STERN, A., 2000. Towards integrating plot and character for interactive drama. In Working notes of the Social Intelligent Agents: The Human in the Loop Symposium. AAAI Fall Symposium, p. 113-118.

NEWMAN, R., 2008. Cinematic game secrets for creative directors and producers. Focal Press, UK.

PASSOS, E.; MONTENEGRO, A.; AZEVEDO, V.; APOLINARIO, V.; POZZER, C.; CLUA, E. W. G., 2008. Neuronal Editor Agent for Game Cinematography. In Proceedings of the VII Games and Digital Entertainment Symposium. Belo Horizonte, p. 91-97.

SPIERLING, U., BRAUN, N., IURGEL, I. AND GRASBON, D., 2002. Setting the scene: playing digital director in interactive storytelling and creation. Computer and Graphics 26, 31-44.

TYAGI S., 2008. A Comparative Study of SVM Classifiers and Artificial Neural Networks Application for Rolling Element Bearing Fault Diagnosis using Wavelet Transform Preprocessing. In Proceedings of World Academy of Science, Engineering And Technology Volume 33, p. 319-327.

VAPNIK, V., 1995. The Nature of Statistical Learning Theory. Springer, New York.

WANG, X., ZHONG, Y., 2003. Statistical Learning Theory and State of the Art in SVM. In Proceedings of the 2nd IEEE International Conference on Cognitive Informatics, p. 55 - 60.

YOUNG, R., 2000. Creating interactive narrative structures: The potential for AI approaches. In Proceedings of the AAAI Spring Symposium in Artificial Intelligence and Interactive Entertainment, Palo Alto, California. AAAI Press.