A neural network approach for real-time collision detection

Ignacio García (1), José D. Martín-Guerrero (2), Emilio Soria-Olivas (2), Rafael J. Martínez (1), Silvia Rueda (1), Rafael Magdalena (2)

(1) Instituto de Robótica. Universidad de Valencia.

Ignacio.garcia@uv.es

(2) Grupo de Procesado Digital de Señales. Universidad de Valencia

Jose.d.martin@uv.es

Abstract-- **The objective of present work has been to develop a Collision Detection algorithm suitable for Real-Time applications. It is applicable to box-shaped objects and it is based on the relation between the colliding objects positions and the impact point. The most known neural network (multilayer perceptron) trained with the familiar backpropagation learning algorithm has been used for this problem; such algorithm models the collision, then decides the impact point and the direction of the forces.**

The algorithm results are very good for the case of box-shaped objects. Furthermore, the computational cost is independent from the object positions and the way the surfaces are modeled, so it is also suitable for Real-Time applications.

The model is being used and validated in a real harbor crane simulator developed by the Robotics Institute for Valencia Harbor in Spain.

Keywords: **Neural networks, real time, simulation, collision detection**

I. INTRODUCTION

Simulation has been widely used for training purposes in the last decades. The benefits that simulators offer are well-known in improving the efficiency of the training process, in lowering the risks included in the trainee, in permitting training in adverse conditions and also in decreasing the costs derived from the use of real devices. The main subsystems that cooperate in a training system are, on the one hand, the instructional design and, on the other hand, the simulator itself. The later is composed of the dynamic model, which decides the interactions among the moving objects; the visual model which represent the virtual scene; the cabin with the controls; the projection system and, optionally, the platform that reproduces the accelerations over the simulator operator. The Robotics Institute in Valencia has developed several simulators [3,12] for SEVASA, the Spanish Society of Stevedores, in Valencia Harbour, one of the most important harbours in the Mediterranean Sea. The simulators are used in the training of the stevedores for the use of Quayside, Gantry, Maffy and Reach Staker cranes.

In real life simulations, not only the dynamic models have to calculate in real time the dynamic behavior of the elements of the simulated environment, but also the interactions among them [4,5,9]. In the case of harbor crane simulators, most of the collisions occur between box-shaped objects, as these cranes manipulate mainly containers. The first algorithms used for collision detection in the simulation system [1] have showed some lacks for the detection of the impact point between boxes and, when their parameters are forced to improve the accuracy their computational cost grows making them unsuitable for the system real time requirements.

In this work it has been presented a new approach based on Neural Networks for the collision detection problem. In the second section it has been explained the problem found when using the usual methods for collision detection. In the third section the proposed method is exposed. In the fourth section some details on the implementation and on the training of the networks have been given.

II. THE COLLISION DETECTION PROBLEM

In simulation and virtual reality environments, the interaction with the scene is fundamental to produce a good immersion feeling in the user. It is very important to reproduce the objects movement with proper accuracy. The physical interaction between the objects should be taken into account and therefore it is necessary to have a good control of such contacts to reproduce the reaction of the system properly, generally by using forces applied to objects. Whenever a collision between a couple of objects happens, the contact point and the direction of the forces must be obtained with sufficient accuracy. Many algorithms have been implemented [1,6,10] in order to obtain both the contact point and the impact direction of the parts that are colliding. These algorithms use polygons to model the object surfaces, and detect the intersections between each pair of polygons applying computational geometry methods. Since testing all possible pairs of polihedra would have an extremely high cost many hierarchical decompositions to accelerate these detections methods have been published [1,7,8,10,11]. However, in some applications a high number of polygons has to be used to determine the impact point and direction with high accuracy. In these cases, as the computational cost of the algorithms highly depends on the number of polygons that are colliding, the mentioned methods can become a bottleneck for the whole system if this number is high.

In the case of collisions between a couple of box shaped objects, despite the apparent simplicity of the problem, the accurate detection of the impact point can overload these algorithms. If the size of the boxes is large, when two planes are almost parallel and are colliding, the intersection can affect a considerable amount of polygons. In these scenarios, a trade off between precision and computational cost must be reached since the quantity of polygons that collide together vary a great deal depending on the position of the objects.

In order to state the computational cost of the surface decomposition, according to the level of resolution needed, the number of polygons to be used is estimated. For simplicity, a couple of identical cubes (that can be easily extended to boxes with different sizes) is considered. As proposed in [1], the faces of the cubes are triangularized. To fit correctly a cube face, firstly it is split into a grid of *n* by *n* squares, and then each square is subdivided into two triangles by its diagonal. The impact region would be determined by detecting the intersections of pairs of triangles. Once the involved triangles are known, the precise impact point can be calculated.

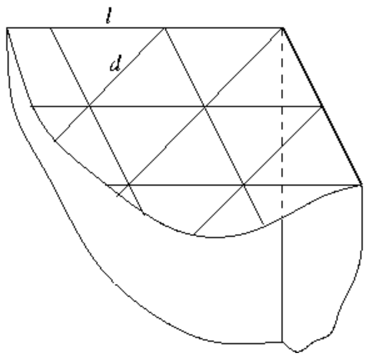

Figure 1. The surface of each cube is subdivided into triangles.

Let us consider a cube with edges of length L. The desired error must be less than a factor *r* of the cube edge *L*, being *r* a small positive number:

$$err \leq rL$$

In order to fulfill with this requirement, the built grid has to be composed of squares with diagonals of length $d \leq err$. As the diagonal of a square of side l is $l\sqrt{2}$, squares with side

$$l \leq \frac{err}{l\sqrt{2}}$$

have to be used and the number of subdivisions in the side of the face will be

$$E\left[\frac{l}{L}\right] \leq E\left[\frac{\sqrt{2}L}{err}\right] \leq E\left[\frac{\sqrt{2}}{r}\right]$$

where $E\,[\cdot]$ denotes the integer part. It can be seen that the number of squares (and therefore of triangles) per face has a growth of order $1/r^2$**.** As a consequence, a relatively high precision leads to a very large number of polygons.

Furthermore, and with high precisions in mind, let us consider a situation in which a vertex of one of the boxes (A) is colliding with a face of the other box (B) with a slight rotation of one object relative to the other (see Figure 2). At the time the collision is detected a small part of the moving object is penetrated into the other box. The intersection of both boxes is a triangle on the face of object B whose size grows as the objects get more parallel, involving a higher number of polygons, and therefore a higher computational cost. Also, in this situation it is more difficult to determine the impact point with the needed accuracy since there is a wide region involved in the collision and further decision techniques should be used to decide the collision point.

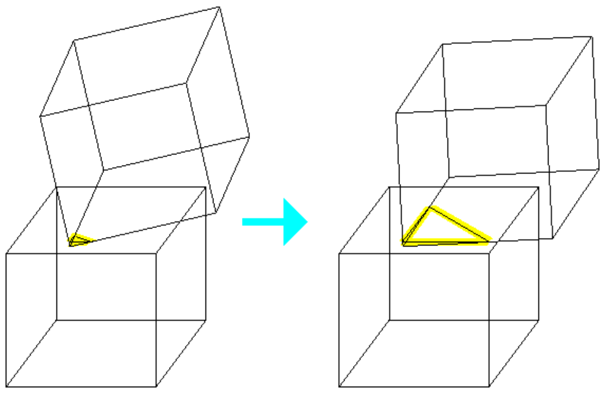

Figure 2. Situation in which the decision of the impact point becomes very difficult by using polygon decomposition of the surface. The shaded region is the region affected by the collision.

III. The Proposed Method

A new method has been proposed based on Artificial Neural Networks (ANNs). A multilayer perceptron (MLP) has been used to model the relation between the relative position and rotation of the pair of boxes and the contact point. This method benefits from the objects geometry in two ways. On the one hand, the geometry imposes constraints in the relation that makes the model easier to be learnt by the Net and, on the other hand, it gives the problem a high degree of symmetry, that allow to divide the problem into several ones with analogous solution. In order to simplify the problem each box has been divided into two trihedra and the function that gives the relation between the position of the boxes and the contact point has been approximated for each trihedron.

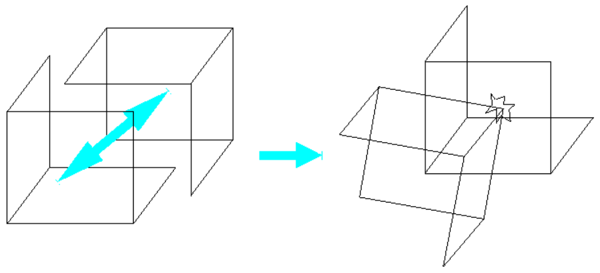

Figure 3. To simplify the problem, each cube has been split into two trihedra and the collisions between two trihedra have been used to generate the data sets.

The model proposed takes as its input the relative position of the two boxes in Cartesian coordinates plus the relative rotation expressed by the Euler angles. The output is the impact point plus a vector that indicates the direction of the collision forces. Note that the length of the vector that points to the impact position can be obtained from its direction, since it must lie on the surface of the cube, and that the vector which expresses the direction of the forces is unitary. The output can then be expressed by the angles of the spherical coordinates of these vectors. Then, the output ranges are [-π,π] for the first and the third output unit and [-π/2, π/2] for the second and the fourth.

IV. Implementation Details

The model has been implemented using an MLP, since this kind of ANNs have been shown to be adequate for solving complex non-linear mappings between two different sets when any *a-priori* knowledge cannot be assumed [2]. In addition ANNs are suitable for real-time applications since they have a fixed computational cost once they have been trained.

An MLP is composed of a layered arrangement of artificial neurons in which each neuron of a given layer feeds all the neurons of the next layer; that is, the network is formed by one input layer, one or more than one hidden layers and one output layer. The first layer contains the input nodes, which are usually fully-connected to hidden neurons and these, in turn, to the output layer [2], as it is shown in Figure 4. Due to the high complexity of the problem explained in Section II, a network with two hidden layers was developed to achieve the desired accuracy in the modelization.

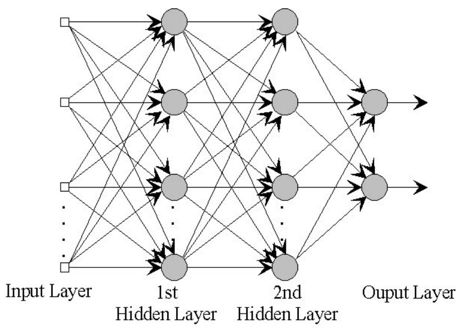

Figure 4. General architecture of a multilayer perceptron with two hidden layers.

The modelization capabilities of this network come from the hidden layers where a non-linear mapping of the input space is carried out. Thus, each neuron of these layers implements a non-linear transformation of its inputs:

$$output_j = \varphi(\sum_{i=0}^{P} w_{ij} \cdot x_i)$$

where the subindex j represents the *j-th* neuron, P is the number of inputs to the neuron, $x_0=1$ is the bias input to the neuron and φ is a non-linear function, typically the hyperbolic tangent. The structure is completely defined by taking x_i to be the external inputs, and $output_j$ to be the output of the neuron.

Since a modelization problem with a continuous output range is being dealt with, it is not necessary the non-linearity in the output neurons; therefore, the function φ is removed in the output neurons, and only the linear combination of the inputs defines the neuron processing.

The training set has been obtained by generating a large set of input positions on the surface of one of the trihedra and considering them as impact points. Then, for each point, each vertex of the other trihedron has been placed on that point. The second trihedron has then been rotated taking a uniform set of values for the Euler angles with the range [–π/2,π/2]. A similar scheme has been followed to obtain patterns for edge-edge collisions. Two boxes of edge 2 have been used to simplify the implementation.

If the mapping that relates the inputs and the outputs of this problem is considered, it can be seen that it is not a continuous mapping. For example, in some cases, a change in the sign of one of the Euler angles can change the impact point from one vertex to another, producing a discontinuity.

To avoid the discontinuities in the training set, it has been divided into subsets with no discontinuities, and the global function has been approximated restricted to each subset. However, both cases, modelization with and without discontinuities, were presented to the neural network. If the results with discontinuities are to be improved, it is possible to develop a committee formed by several networks, where each one is specialized in one different zone of the input space.

Over 2,000 networks have been trained to tune the parameters of the network (adaptation and momentum constants, number of hidden nodes and amplitude of the initial synaptic weights), thus obtaining the best performance models for both problems, with and without discontinuities. The backpropagation algorithm has been used to train the network and every training was stopped by cross-validation. Three sets of data were used in both problems analyzed in this work: a training set, a validation set (used for cross-validation) and an external test set. In the problem with discontinuities, the training set was formed of 8,000 patterns, the validation set of 5,000 and the test set of 75,000. In the problem without discontinuities the training set was formed of 10,000 patterns, the validation set of 6,000 and the test set of 50,000.

V. RESULTS

The best results for the model with discontinuities (Table 1), have been obtained within a multilayer perceptron with architecture 6×20×18×4. The models have been evaluated through three performance indexes: the Mean Error (ME), as a measure of bias, and the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE) as precision measures. Keeping in mind the range of the inputs (the average is over 4.25), it has been observed that the average error in a single pattern is near 10 % of the range, and looking at the ME, no bias in the modelization has been observed. It is worth noting that the errors made are very similar in the 4 outputs. This result is good enough to perform a response after the collision.

Table 1. Best results obtained with a multilayer perceptron for the model with discontinuities.

	Training	Cross-validation	External validation
ME	0.0332	0.0259	-0.0005
MAE	0.4161	0.4383	0.5624
RMSE	0.6947	0.7363	0.8667

Figure 5 shows the modelization for the first of the four outputs in part of the external validation set. The solid line represents the desired output and the dotted values represent the corresponding outputs provided by the model.

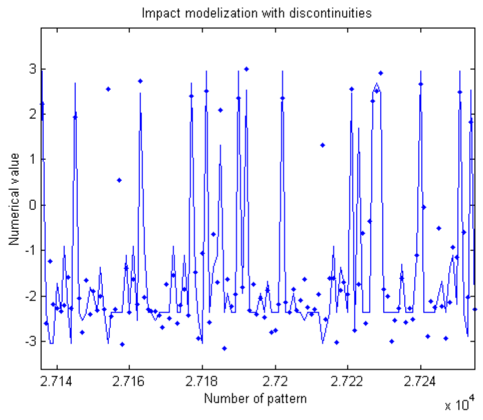

Figure 5. Prediction in the test set for the first output of the problem with discontinuities. The solid line represents the desired output and the dots the corresponding outputs provided by the model.

For the model without discontinuities, the results have been obviously better. The architecture of the best network was 6×20×17×4. The obtained results are shown in Table 2 and Figure 6. Once again, the errors made in every output were balanced, the average error was below 10 % of the range of the outputs and no bias appeared in the modelization.

Table 2. Best results obtained with a multilayer perceptron for the model without discontinuities.

	Training	Cross-validation	External validation
ME	-0.0008	0.0031	0.0467
MAE	0.2765	0.2883	0.4523
RMSE	0.5410	0.5748	0.7302

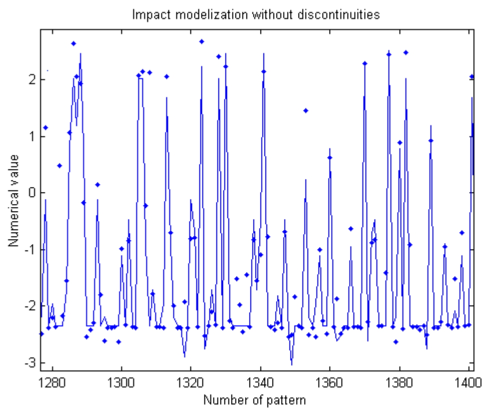

Figure 6. Prediction in the test set for the first output of the problem without discontinuities.

VI. CONCLUSIONS

The work presented shows a new approach to the problem of collision detection suitable for Real Time simulations of environments where most of the objects have a similar shape. On the one hand, the results obtained up to the moment indicate that they fit the needs to obtain the impact point and direction accurately. On the other hand, the method has a fixed computational cost since it does not depend on the number of polygons used to model the objects but only on the number of units of the neural network used in the model. In addition, the well-known versatility of methods based on Neural Networks eases its application to other environments.

VII. FURTHER WORK

The complete collision detection system works in two steps: initially, a multilayer perceptron or another classifier performs a classification based both, on the properties of the relative position of the objects and on its colliding parts (vertex, edges and faces). Then, for every class, a new MLP decides the problem outputs (the impact point and direction). This expert committee allows an important knowledge on each part of the input space; thus, the different outputs of the models are combined, and a more accurate system can be achieved in the future.

Once this method is completely implemented, a set of tests will be carried out to compare the results with those of the methods existing in literature. However, in a more precise way, the validation that has been done with large data sets shows that the error is properly controlled.

VIII. References

[1] S. Gottschalk, M. C. Lin, and D.Manocha. OBBTree: A hierarchical structure for rapid interference detection. *Computer Graphics, 30 (Annual Conference Series):* 171-180, 1996.

[2] S. Haykin. *Neural Networks – A Comprehensive Foundation*. Upper Siddle River, NJ: Prentice Hall, 1999.

[3] M. Lozano et al. "A reconfigurable projection system for several gantry crane simmlators". *Third International Immersive Projection Technology Workshop,* pages 167-177, 1999.

[4] V. J. Milenkovic and H. Schmidl. Optimization-based animation. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 37-46, 2001.

[5] S. Redon, A. Kheddar, and S. Coquillart. Contact: Arbitraty in-between motions for collision detection. *Proceedings of IEEE Workshop on Robot-Human Interaction, ROMAN'2001,* 2001.

[6] F. Thomas et al. Computing signed distances between free-form objects. *In Proc IEEE Int. Conf. On Robotics and Automation (San Francisco (CA)),* 2000.

[7] G. van den Bergen. A fast and robust GJK implementation for collision detection of convex objects. In *Journal of Graphics Tools*, vol 4 n 2, 1999.

[8] G. van den Bergen. Efficient collision detection of complex deformable models using AABB trees. *Journal of Graphics Tools*, (4):1--13, 1997.

[9] D. Baraff and A. Witkin. Dynamic simulation of non-penetrating flexible bodies. *Computer Graphics*, 26, 303--308. 1992.

[10] M. Lin and S. Gottschalk. Collision detection between geometric models: A survey. *Proc. of IMA Conference on Mathematics of Surfaces*. 1998.

[11] Wilson, E. Larsen, M. C. Lin and D. Manocha. IMMPACT: Partitioning and Handling Massive Models for Interactive Collision Detection. *Proc. Eurographics'99*. 1999.

[12] F. J. Seron, M. Lozano, M. Fernández et al. Una aplicación de los entornos virtuales al mundo de la simulación civil. Actas Ingegraf' 99. 1999