# Interactive Projection Game

[CSEE 4840 Project Proposal - February 2008]

Abdulhamid Ghandour
ag2672@columbia.edu

Thomas John
tj2183@columbia.edu

Jaime Peretzman
jp2642@columbia.edu

Bharadwaj Vellore
vrb2102@columbia.edu

## 1. INTRODUCTION

The goal of this project is to design a system to create a virtual pool-like game using vision and projection technologies.

Game play is based on a projected image of a pool-table-like surface, with a ball positioned on it. A player can then use a cue or cue-like object to 'strike' the ball. The ball is then projected in the direction it was struck and made to settle at a new final position from where the player can strike it once again.

The detection of the 'strike' is done using a camera which captures the projected image with the cue stick over it. The image is then processed to determine the direction and speed of the movement of the cue tip relative to the position of the ball. The data gathered from this processing is then used to compute the trajectory and distance of motion of the ball, if any, and reposition the ball appropriately. As the ball moves and is repositioned, new images of the table and the ball are redrawn and projected for the player to be able to admire his stroke and plan his next one.

The image capture and vision-based input is processed entirely in two dimensions. The implication of this choice is that the angle of impact of the cue on the ball cannot be judged or considered when processing the movement of the ball.

## 2. HIGH-LEVEL DESIGN

This section details a first-cut high-level data flow digram together with a list of components that are anticipated to be incorporated in the design of the system.

### 2.1 Parts List

- Digital camera 1.3 Mega Pixel TRDB-DC2
- DE2 Altera Development Kit
- Standard Projector with VGA input

### 2.2 Block Diagram

Figure 1 shows a block level representation of the system.

- Projector - We use an external projector which displays images of the table with a ball placed on it. The image of the table is created with a high concentration of saturated green so as to simplify the detection and processing of the black cue stick over this background colour. The projector receives data from the board via VGA.
- Camera - The TRDB-DC1 digital senses the image displayed by the projector together with any shape introduced over the projected image.
- Vision System - The vision system consists mainly of three elements (these are implemented in hardware):
  - Camera Interface - This component is in charge of setting the initial configuration of the camera every time the system is started up. These settings may include such parameters as the desired resolution, the number of frames that are desired to be captured per second, and the gain for each colour. The interface is in the form of a serial interface to the camera which is understood and implemented by a custom component on the FPGA.
  - Frame Grabber - This is a component that captures 10-bit RGB values from the camera into a frame buffer by working at the appropriate pixel clock frequency. A double buffering scheme may be useful to enable processing a frame while the next frame is being captured and stored in memory. The choice of the amount, if any, of memory that will be employed to hold one or more frames is based on the algorithm that is chosen for parsing the image to detect objects. At the moment, the exact algorithm, and consequently, the memory requirement are not entirely clear. On the basis of a preliminary algorithm that has been implemented and tested in a desktop environment, it is anticipated that one complete frame may be required to be held in memory for processing. The memory to be used is the SRAM memory. This decision is based on the superior speed on the SRAM.
  - Vision Algorithm.- The vision algorithm scans the captured raw image to detect the tip of the cue

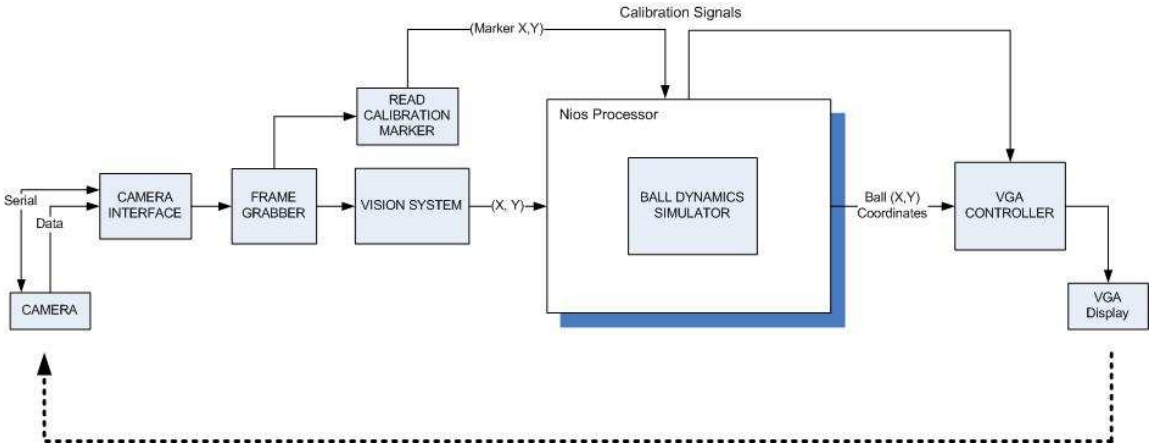


**Figure 1: Block Diagram**

stick and return the 'x' and 'y' co-ordinates of the tip. Sample results from a simple implementation of an algorithm for this are included later in this document.

- Real Time Object Dynamics Simulator - The algorithm, implemented in software, calculates the position and movement of any ball in the screen. This computation is based on the 'x' and 'y' co-ordinates and the 'x' and 'y' velocity vectors, which are determined from impact information received from hardware. Damping factors and collisions with borders are also incorporated in the algorithm to determine the placement of the ball.
- Frame Generator - This module refreshes the frames to be emitted by the projector. This is implemented in software and involves configuring a display controller with information regarding the location of objects to be drawn on the projected image.
- Display Driver - This last component in the chain generates the signals for the VGA such as active video, blanking and synchronization. This is a physical layer signal generation block implemented in a VHDL module.

## 2.3 Sample Results from Image Scanning Algorithm

Figure 2 and Figure 3 show preliminary results from an algorithm that searches a bitmap for the presence of a cue stick. The image background is designed to be entirely green to simplify object detection based on colours that are observed in the captured bitmap.

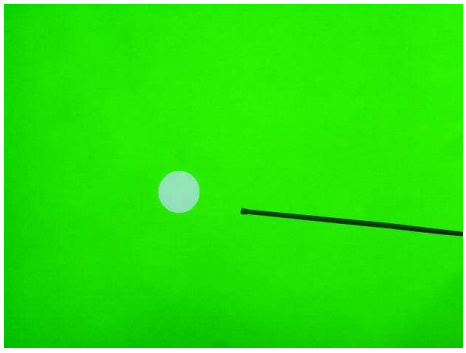

**Figure 2: Cue Stick Tracking Example**

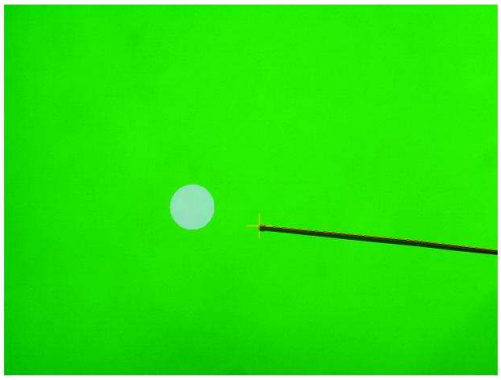

**Figure 3: Cue Stick Tracking Example - With cross-hair at cue tip**

# 3. RISKS AND CHALLENGES

We foresee the following technical risks in implementing this project:

1. Input frame processing - The precise algorithm and the corresponding memory requirement in terms of frame buffers for the input is not yet clear. The speed of this processing is important as it could potentially affect sensitivity to the movement of the cue stick.

2. Noise - The impact of noise in the incoming data and the sensitivity of the camera is expected to impact the algorithm and its tuning.

3. Calibration - The system is sensitive to the position and orientation of the camera and the projector, and therefore, changes in these elements can impact the calculation of co-ordinates of various objects that are drawn and detected.

## 4. MILESTONES

1. Milestone 1
    - Hardware implementation of the camera interface
    - Hardware implementation of the object detection algorithm through pixel scanning
    - Test-benches for the hardware implementation, and demonstration of image capture and pixel analysis through a simple application that indicates the co-ordinates of the object detected on a VGA monitor.
2. Milestone 2
    - Thorough calibration of the system to fine-tune the projection and camera modules and achieve accuracy in determining object location.
    - Basic implementation of the software bouncing ball
3. Milestone 3
    - Optimization of hardware
    - Full development of software design, and testing
4. Final Milestone
    - Testing with a projector
    - Report and presentation completion