Aperture Bioscience Laboratories

Progress Report 9 for NICView: A Virtual NICU Simulation

Due: 4/3/15

Professor Walker

Amy Young, Caitlin Li, Jennifer Duan, Pamela Wu, Lindsey Sumners

I. Background of the project

Considering the new regulations that limit work hours for medical residents, these residents do not have the same real case experience as in the past. Therefore, this reduces patient safety, because the residents do not experience a large and varied amount of cases. In order to solve this problem, our project will involve developing a simulation game that can be played at the resident's home. By playing this game, the residents will be able to gain better conceptual knowledge by going through different scenarios, which addresses the issues of volume and variety of cases and patient safety. By having this knowledge, the residents will be able to use the time in the hospital in order to gain experience with the physical actions in medicine.

II. Achievements since the last report

The main area of progress has been in making the medical inventory and having it become functional. The sprites that will be dragged to the baby have been made, which now makes the inventory interactable. The code is shown below.

```
//INVENTORY

using UnityEngine;
using System.Collections;
using System.Collections.Generic; //gives access to List type

//create variables
public class Inventory : MonoBehaviour {
	public List<Item> slots = new List<Item>(); //list of items in slots
	public int slotsX, slotsY, xpos, ypos, xsize, ysize, offset; //number of slots and position and size of slots
	public int bxpos, bypos, bxsize, bysize; //position and size of button
	private bool showInventory; //whether or not inventory appears
	private bool showTooltip; //whether or not to show pop up box when hovering over item in inventory
	private string Tooltip; //text that shows up in said box
	public Item CPAP = new Item ("CPAP", "Continuous Positive Airway Pressure", Item.ItemType.Airway); //CPAP mask
	public Texture tool = Resources.Load<Texture2D> ("tool"); //background image for inventory
	public List<string> ItemNames = new List<string>(); //list of item names
```

```
	public List<GameObject> Devices = new List<GameObject>(); //list of item
game objects

	string CreateTooltip(Item item) //create the text in the tooltip box
	{
		Tooltip = "<color=454545>" + item.itemName + "</color>\n\n" +
"<color=473344>" + item.itemDesc + "</color>";
		return Tooltip;
	}

	//Populate slots with empty items
	void Start ()
	{
		ItemNames.Add ("CPAP"); //creating a list of item names
		for (int i = 0; i < (slotsX * slotsY); i++) //looping through number of slots
		{
			slots.Add(new Item()); //creating a list of empty items as long as
number of slots
		}
		AddItem (CPAP); //Add CPAP to slots
	}
	//Activate or deactivate item gameobjects depending on if show inventory is true
	void Update()
	{
		for (int i = 0; i < slotsX*slotsY; i++) //loop through number of items
		{
			Devices[i] = GameObject.Find(ItemNames[i]); //find the game
object with that name
			Devices[i].SetActive(showInventory); //set it to the same state as
the inventory
		}
	}
	void OnGUI()
	{
		if (GUI.Button (new Rect (bxpos,bypos,bxsize,bysize), "Crash Cart"))
		{
			showInventory = !showInventory;
		}
		Tooltip = ""; //set text to empty
		if (showInventory) { //if show inventory is true, draw the inventory
			DrawInventory ();
		}
		if (showTooltip)
		{ //if showtooltip is true, draw a box containing the tooltip
			GUI.Box (new Rect (Event.current.mousePosition.x + 15f,
Event.current.mousePosition.y, 200, 200), Tooltip);
```

```
		}
	}

	void DrawInventory() //define draw inventory
	{
		int i = 0;
		for (int y = 0; y < slotsY; y++)
		{
			for (int x = 0; x < slotsX; x++) //loop through indices of grid
			{
				Rect slotRect = new Rect(xpos+x*xsize+offset,
ypos+y*ysize, xsize, ysize); //rectangle
				GUI.Box(new Rect(xpos+x*xsize+offset,
ypos+y*ysize,xsize,ysize),tool); //background image
				if(slots[i].itemName != null) //if the slot is not empty, draw
the item icon in the corresponding box
				{
					GUI.DrawTexture(slotRect,slots[i].itemIcon);
					if(slotRect.Contains (Event.current.mousePosition))
//additionally if the mouse is over it, show tooltip
					{
						CreateTooltip(slots[i]);
						Tooltip = CreateTooltip(slots[i]);
						showTooltip = true;
					}
				}
				if(Tooltip=="") //if the tooltip is empty, don't show the
tooltip box
				{
					showTooltip = false;
				}
				i++;
			}
		}
	}

	void AddItem(Item device) //create a method for adding items to the inventory
	{
		for (int i = 0; i < slotsX*slotsY; i++)
		{
			if(slots[i].itemName==null)
			{
				slots[i] = device;
				break;
			}
		}
```

```
        }
    }

    //AwfulMedia inventory tutorial, but inventory in game will never change
    //so reduced everything to only slots

    //ITEM

    using UnityEngine;
    using System.Collections;

    [System.Serializable] //makes it so that all attributes are listed under item
    public class Item {
        public string itemName;
        public string itemDesc;
        public Texture2D itemTexture;
        public Sprite itemIcon;
        public ItemType itemType;
        public int iconx,icony,iconw,iconh;

        public enum ItemType {
            Airway,
            IV,
            Medication,
            Other
        }
        //two constructors for creating items: one with attributes and one empty
        public Item(string name, string desc, ItemType type)
        {
            itemName = name;
            itemDesc = desc;
            itemIcon = Sprite.Create (Resources.Load<Texture2D>("Icons/" +
name),new Rect(iconx,icony,iconw,iconh),new Vector2(.5f,.5f));
            itemType = type;
            itemTexture = Resources.Load<Texture2D> ("Icons/" + name);
        }
        public Item()
        {

        }
    }

    //AwfulMedia tutorial, only removed item ID designation and changed types
```

III. Deviation from the plan and corrective action

Looking at the work plan, the project is behind in one main area, which is the timer. The team has been working on debugging the code that was shown in Progress Report 7. However, since the other parts of the scenario have been completed, the team can focus on this aspect of the project and finish programming the second scenario.

IV. Plan for Next Week

Over the next week, the team plans to focus on debugging the timer code and integrating it into scenario two. The third scenario will then be programmed using the medical inventory, timer, and points code that has already been developed. Finally, the team will set up the online forum and consult Dr. Krakauer on the testing protocol for the game. This will allow the team to know when the game is tested and how many residents will play it.

V. Assessment of Progress

Considering the current progress, the team is a little behind schedule. However, by accomplishing the goals set out for next week, the team will be able to finish the project on time.