Isometric Demo Design Document

Jason Richardson
www.jasoneldred.com\GameDesign\
m.jason.richardson@gmail.com
(256) 508-0353

1.0 The Demo

1.1 Abstract

This document describes a concept demo for a single-player, 3D isometric-viewpoint, action RPG. The limited scope of this demo will not allow for exposition of the intended RPG elements, but instead will focus on the fast paced, run-and-gun combat that will be core to the gameplay, a unique combat system which will help the game break out of the mold of isometric Diablo clones.

1.2 Using UDK

This demo will be implemented as a mod of Epic Games' Unreal Development Kit (UDK, http://www.udk.com/). Because this demo is an exercise in prototyping, no unique assets will be developed (with the exception of a simple game level). Instead this demo will reuse existing UDK meshes, textures, sounds, etc.

2.0 The Gameplay

2.1 The Camera Angle

The camera angle will be isometric, a view with the camera oriented at 45 degrees around the vertical (Z) axis, and then pitched downward so that the horizon is out of view. The view will at all times be centered on the player's character. Because of 3D perspective, the player will be able to see more into the distance at the top of the screen than at the bottom. To prevent this from becoming a balance issue (with the player able to see enemies before they can see him, for instance) the camera should be pitched downward at an angle greater than 45 degrees. The camera's field of view (FOV) should also be small, probably 60 degrees or less.

The player will be able to zoom in and zoom out on the isometric view within bounds. At maximum zoom level, the player character should fill about half of the screen's

vertical height. At the furthest zoom level, the player should be able to see just beyond the range of his weapon.

2.2 The Game Level

The game level will be simple, an indoor, dungeon-like environment consisting of an entrance and a corridor that forms a rectangular loop leading back to the entrance with nooks and crannies along the way for interest. There should be a few obstacles placed within the corridor to help test pathfinding. As an optimization, this level will be built out of tiles and meshes that are smaller than the visible screen space. This will allow the engine to frustum cull level geometry when it leaves the field of view. For simplicity this demo's game world will be vertically level. There will be no slopes or stairs to provide vertical depth. The basic layout of the level should be as shown in Figure 2.2 below:

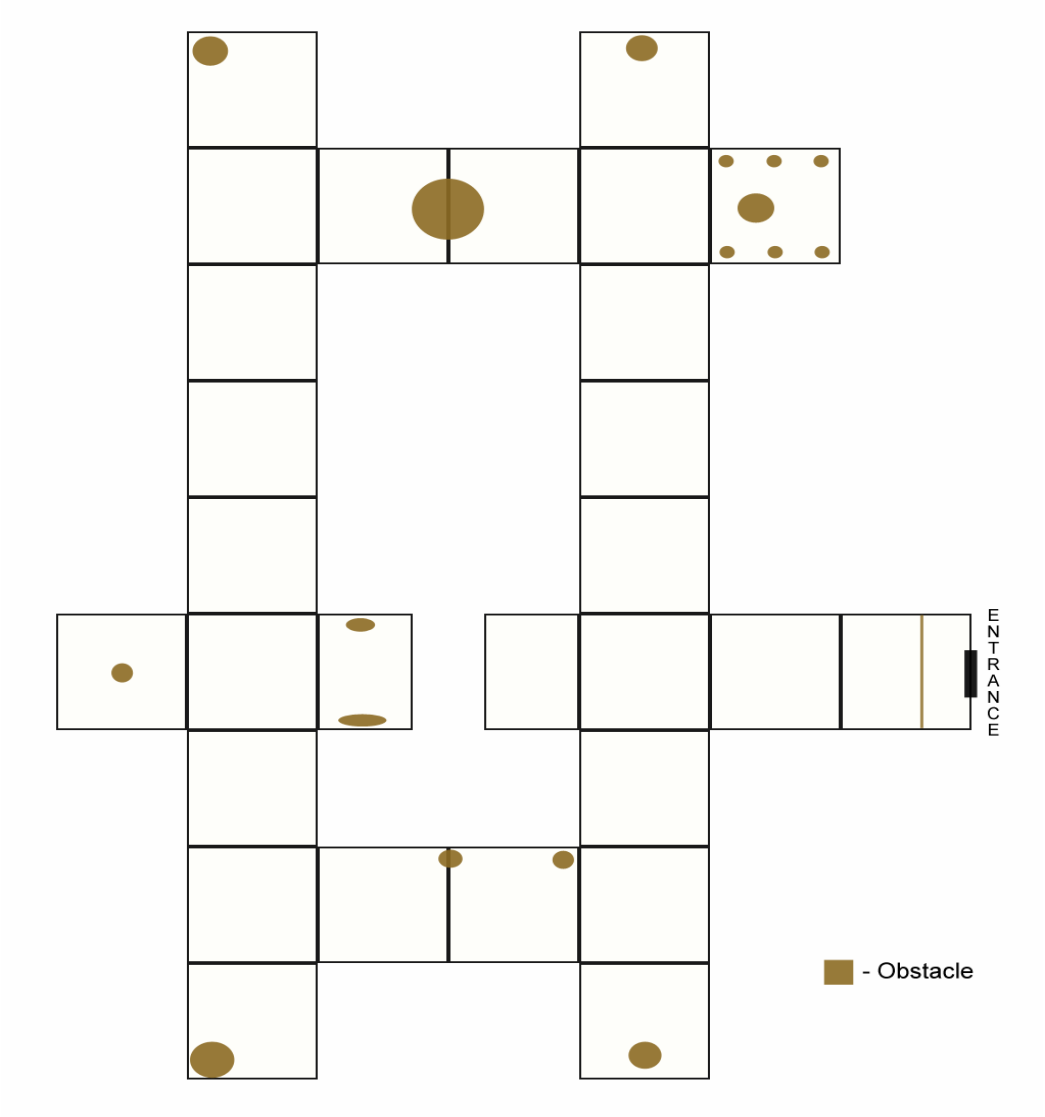

Figure 2.2

2.3 Movement

Similar to the movement system common to isometric games, players will be able to target a location with the mouse cursor and click a “Movement Button” to make the player character (PC) run to that location. If the target location is in the line-of-sight of the PC, the PC will move directly there. However, if the PC’s view of the target location is obstructed, the game will use UDK’s Navigation Mesh pathfinding to allow the PC to find his way to the location.

When the player has depressed the Movement Button for a sufficient duration, the game will interpret the input as a “hold” rather than a “click” and will not perform the pathfinding mentioned previously. Instead, the PC will begin to run continuously in the direction of the mouse cursor for as long as the Movement Button is held.

This game will offer a further movement control, unique from other isometric games. When the PC is already moving (either because the player “clicked” or is “holding” the Movement Button), the player will be able to press and hold an “Auto-Run Button” (such as the SHIFT key) to lock the PC into running in his current direction even if the Movement Button is released; the PC will continue to run until the Auto-Run Button is released. The usefulness of this control will be exposed in the next section.

2.4 Combat

Unlike in other isometric games, here the player will not control movement and attacks with the same button. Instead clicking a separate “Attack Button” will dictate that the PC should fire his weapon in the direction of the mouse cursor. If the player clicks on a hostile non-player character (NPC), then the weapon will aim for the center of the target (a location likely determined by a bone in the NPC skeletal mesh). If the player does not click on an NPC, but instead clicks out in the world, then the weapon will fire horizontally in that direction; it will not target the precise location underneath the mouse cursor.

When making an attack, the attacker (either the PC or an NPC) will turn toward the direction it is attacking. If this is different than the direction it is currently moving, then the attacker will be shown back-pedaling or sidestepping (as appropriate) in the direction that it was already moving while facing its attack target. By employing the “Auto-Run Button” described in the previous section, the player will be able to lock the PC’s movement direction and then use the mouse to shoot at the same time as the PC is sidestepping or backing away from opponents.

For a brief time after making an attack, the attacker should continue to face toward its target. This is to prevent the attacker from turning rapidly back and forth between its attack direction and its movement direction.

Note: this demo provides only ranged combat.

2.5 Non-Player Characters

AI will be simple. If the player enters an NPC's field of view (180 degrees in front of the NPC, within a set distance), if the player gets too close behind an NPC, or if the player makes a sound (such as firing a weapon) within hearing range, then the NPC will select the player as a hostile target. At that point the NPC will be capable of two behaviors: following and attacking.

NPCs following the player will use a similar system to what the PC uses to follow the mouse cursor. The NPC will move directly toward the PC if he is in line-of-sight. Otherwise the NPC will use nav-mesh pathfinding to get as close as possible to the PC's location.

NPCs will have a maximum range at which they can aim and shoot toward the PC. This range should be less than the range necessary for the NPC to see the PC in front of them. This will allow NPCs to be alerted to the PC's presence from offscreen without being able to shoot at the player from offscreen.

The game will force NPCs to hesitate for a short, random amount of time after performing an attack to prevent their attacks from occurring at regular intervals. NPCs will also be subject to accuracy penalties. The game will compute a random yaw and pitch offset (from ranges specified for the NPC) to apply to the NPC's shot. This will prevent the NPCs from being perfectly accurate and will help the player survive attacks from multiple opponents.

2.6 Death and Respawn

To be entertaining, both NPCs and the PC will use ragdoll physics when dying to produce realistic falling animation. Ragdoll bodies will be interactable and can be shot or trampled.

For this demo both the PC and the NPCs will respawn after dying. The PC will have a short delay after dying before he will automatically respawn. NPCs should have a longer respawn delay to prevent overwhelming the player with opponents. The specific respawn delay for the NPC should be configurable from inside its editor-placable spawn point, so level designers can customize respawns.

2.7 Stats and Visuals

The player will use UDK's "Iron Guard" skeletal mesh and will have 400 hitpoints.

This demo will feature only a single type of enemy NPC using UDK's "Cathode" skeletal mesh. NPCs will be given a random number of hitpoints between a given min and max. In this case the range will be 18-23 hitpoints.

The game will feature only a single weapon. With no original assets available, this demo will reuse the mesh, particle systems, and sounds of UDK's "Shock Rifle". The weapon will be semi-automatic, meaning that the player will have to click once for each shot. The shock rifle beam will be shown emerging from the weapon's muzzle and terminating at the computed attack location. This custom shock rifle should have a knockback effect. It should be able to be fired somewhat more frequently than every half-second, should have a range of approximately 40 feet, and should cause 20 points of damage per hit.