Virtually Learning to Sail

By:
Cody Steward

Supervising Committee:
Ben DeVane (Chair)
Angelos Barnpoutis
Peter Ifju

Table of Contents

Abstract

This paper serves as a written complement to my project in lieu of a thesis (otherwise known as "thesis project"). It will describe the motivation to create a digital tool for teaching principles of windsurfing and the feasibility of this project. The abilities and limitations are addressed as well as the potential benefits of the project. The postmortem summary of this project is included as well.

Introduction

Serious games are constantly being developed into increasingly better teaching tools (Gee et al., 2005). Additionally, virtual trainers are being used in many fields from the military to medicine (Rice, 2012). I propose we bring this medium to athletics. Athletics have always been a difficult subject to teach. There is no substitute for "learning by doing" in the case of athletics, but sometimes there are limitations that prevent participation. If a student is injured, or is in a location where a certain athletic activity is not possible, a classroom setting has to be used. A virtual trainer would bridge the gap between the classroom and first hand participation.

Virtual training would also make the learning process of athletics much safer, especially in extreme sports and sports that have a high amount of risk due to their complexity. Some sports, such as sailing, can be both extreme and complex. Athletes are away from the controlled environment of solid land and are at the mercy of the elements. Wind and water are two extremely unpredictable and uncontrollable forces that can change very quickly and without notice (Coutts & Dornellas, 2001). A virtual trainer would allow for a completely controlled and safe learning environment for athletes.

Original Proposal & Design Framework

I propose to create a virtual trainer for windsurfing. The skill of sailing applies to several individual sports, and once mastered can be applied to all. I have been a US Sailing Windsurfing Instructor for ten years and have seen the positives and negatives of the current instructional methods. I also have worked for several companies developing military training simulators, and have been studying video game theory and design for the past few years. These influences have inspired me to bridge the gap of classroom learning and on the water learning for windsurfing with a virtual trainer (serious game).

I propose to teach the points of sail through a first person video game experience. I will use Crytek's free source development kit (SDK) license of CryENGINE 3 (CE3) to create this virtual teaching tool. I have chosen CE3 due in large part to its aesthetics and ability to handle large outdoor environments (Crytek, 2012). Realism will be the key to conveying the abstract ideas of wind and I feel that CE3 is the best software for creating this straight off the shelf (Figure 1). I believe realism is an important element for this tool because I will be teaching and demonstrating abstract concepts. If these abstract elements are presented as close to reality as possible then the effort needed to believe them will be minimal (Squire, 2006). The free SDK also has a strong support base at Crydev.net. Crytek employees visit the forums often to help people as well as listen to user input to improve features. I have also begun to foster a relationship with Crytek representative's at Real Time Immersive (RTI) in Orlando. They have shown interest in this project and have offered to help me with learning CE3.

Points of sail is a universal sailing theory that applies to all sailing sports. It revolves around

reading the direction of the wind and sailing on an angle in reference to that wind (Coutts & Dornells, 2001; Jones 1987). It works much the same as locating north with a compass and hiking at an angle relative to the compass' readings. I propose to combine elements of the current paper-based instruction on points of sail and the experience of first person simulation to create a single tool that provides both of these teaching methods at once. This tool will be designed for windsurfing but the lessons and mechanics will be applicable to all sailing crafts. The user will control a windsurfer in first person view within CE3. The view will be the same as real life, with the user's view free but the body restricted. This will provide a similar "feel" to real life windsurfing. A head's up display (HUD) in the game will display similar graphics to the diagrams that are used in the current instructional book material. A top-down view of the windsurfer will be shown in a compass-like setup that has "north" as the wind direction. The sail will move depending on the user's input for steering, and the board will turn as a reaction to the sail control. This top-down view is the current visual tool for teaching points of sail on paper. HUD's are a common feature in most video games. Users are accustomed to filtering the information given by the HUD while playing the game. Combining this with the realism of a first person experience will help the user to process all of the different information involved in the points of sail. This setup will engage the user in a reading-while-doing method of learning.

Technology

CryENGINE 3

Crytek's free SDK is a full and unrestricted license for their game engine software, CryENGINE 3. It is one of the most advanced game engines currently on the market and has a constantly growing user base. Crytek is also aggressively branching outside of video games and is involving their engine with the film industry, through Cinebox, and the military, through RTI (Hill, 2012). This acceptance in multiple fields makes it a great choice for photo realism and teaching applications. Points of sail is a rather abstract concept and arguably only applies to sailing sports. Therefore representing it realistically, as opposed to stylistically in the game art, is a high priority. CE3's assets are all rendered realistically making it much easier to suspend disbelief than if the assets look cartoony or stylized.

Figure 1. A real world location recreated in CryENGINE.

CE3 is also designed with large outdoor environments in mind (Figure 2). Sailing is an outdoor sport and I intend to recreate the outdoor experience for this trainer. CE3's ability to handle large amounts of geometry and effects outdoors makes it a perfect choice. A beach setting with lots of vegetation would be more difficult to achieve in other engines. There are also many assets included with the free SDK that are perfect for such an environment, thus saving on production time.

Figure 2. An example of CryENGINE's exterior evironments.

Some assets will need to be created in a third party digital content creation (DCC) software package. I have the most experience with DCC in Adobe Photoshop and Autodesk 3D Studio Max (3DS Max). Crytek favors these applications and so they have excellent integration with CE3. This will make creating external assets much easier and decrease the amount of time learning more tools to create the unique assets I will need (e.g. a windsurfing rig and board). Photoshop can create textures and export them as a special crytiff file, and 3DS Max can export both models and animations in a .cdf file format.

Since CE3 is made to handle outdoor environments it is also well equipped to handle water. This is beneficial because boats and features like buoyancy are already included in the engine and will not require custom scripting. For elements that will require scripting, CE3 uses Lua as its primary scripting language. Lua is based on the C++ language and therefore has a massive support base. I am very interested in collaborating and leading a team for this project and as such I have recruited the help of two capable software engineers. However, this will be an extremely minor part of the overall project as my main focus is on the teaching impact of this tool, not the tool itself, and the need for custom scripts is extremely limited. At this initial state only one custom script is needed and it is not very complex. A custom HUD will also be needed. I am interested in both teaching others with this tool and leading the next generation of teachers. As such I feel it would be invaluable to incorporate some teamwork elements into the production process of this trainer. CE3 has many tools for handling production across teams and will help facilitate this process (Keith, 2010).

Since Crytek's release of the free SDK, and their successful efforts to branch into cinema and military simulations they have developed excellent support (Hill, 2012). Crydev.net is full of everything from brand new users like me to on-staff developers from Crytek. The free SDK's abilities are nearly unlimited, and as such many experienced users of CE3 have transferred from the modding community to be able to create content that is not limited to the video game, Crysis 2. The forums are constantly populated with people of all skill levels giving input (Crytek, 2011). I have also begun to form a relationship with RTI, Crytek's representative company in the industry of military training

(Figure 3). They are based in Orlando and have shown some interest in offering me support on this project if needed. They are certified CE3 instructors and licensed distributors. If the free SDK becomes limited for what I need, it will be possible to buy a research license through them.

Figure 3. A screen shot from one of RTI's military simulations.

My experience as an instructor for ten years shown me that teaching can be tough. The most difficult part of teaching is finding out how the students learn. There are many different types of learning styles: kinesthetic, auditory, visual, literary. Each student has strengths and weaknesses and capitalizing on those creates the best learning experience. Teaching a sport demands a strong kinesthetic learning experience. Some students, however, benefit more when I sit them down and diagram the concepts on a whiteboard. Others learn best through my verbal descriptions. Some do best by watching me, and of course plenty learn by just doing the activity. The problem is that I cannot deliver all of these teaching methods at once. Each has to be presented individually.

Video games are close to being able to present all of those ideas at once. Introduce a student to a video game and they are looking, listening, and reading. They are even "doing" to some degree in the form of controls, be it a keyboard and mouse or a console controller. Knowing what it takes to teach sailing in real life and the broad range of learning types, I can see the benefit of putting this sport into a video game environment. Students can be told what to do through a non-player character (NPC) instructor (auditory). They can read instructions from HUD popups or info panels (literary). They can see the results of their actions from the first person view, or watch a NPC (visual). All the while they are engaging their body as they manipulate the controls of the game.

This form of kinesthetic learning is no substitute for the real sport. However, it is perfect for creating a bridge between the instructor's classroom and hitting the water. Combining all these teaching methods into one form would make the delivery of important concepts, in this case points of sail, much easier and less time consuming. Since the points of sail is such an abstract concept it is often handled in different ways. Commonly it will be summed up quickly so that a student will have enough knowledge to "survive" and enjoy themselves. Otherwise it is be explained in depth and will take a very large amount of time because it is a complex concept. Combine this with the difficulty of figuring out a student's learning type this is a hefty dilemma to tackle as a teacher. Creating a multi-sensory learning experience would be the key to providing both depth and speed for teaching. Any student could be presented the information and find their best source of learning from it. With the proliferation of digital devices today, especially personal computers and consoles, a video game is a

perfect way to distribute this method of multi-sensory teaching (Schmorrow, 2012).

Sailing

Points of sail is a universal concept across all sailing sports. It is one of the basic models that all sailors must learn if they want to be successful in the sport. The terms and concepts are used in normal sailing conversation. Runners might say "I almost had you on that last lap," but sailors would say, "I almost caught up to you on the beam reach, but you pulled away from me on the run." Whether this concept is taught directly or indirectly, points of sail is vital to sailing. Those who understand it the best often make the best sailors and competitors. Knowing when to close haul and why can make all the difference in a race. The type of board I race on is fastest on a beam reach. Without that knowledge I would be at a serious disadvantage on the race course. Points of sail is the sailor's way to orient themselves to the wind. This is the most important concept and yet the hardest concept to grasp in sailing. How do sailors orient themselves to an unseen force? People often orient themselves to the shore since that is "home" or "safe" but in reality the only orientation is to the wind. Land has no effect on sailing.

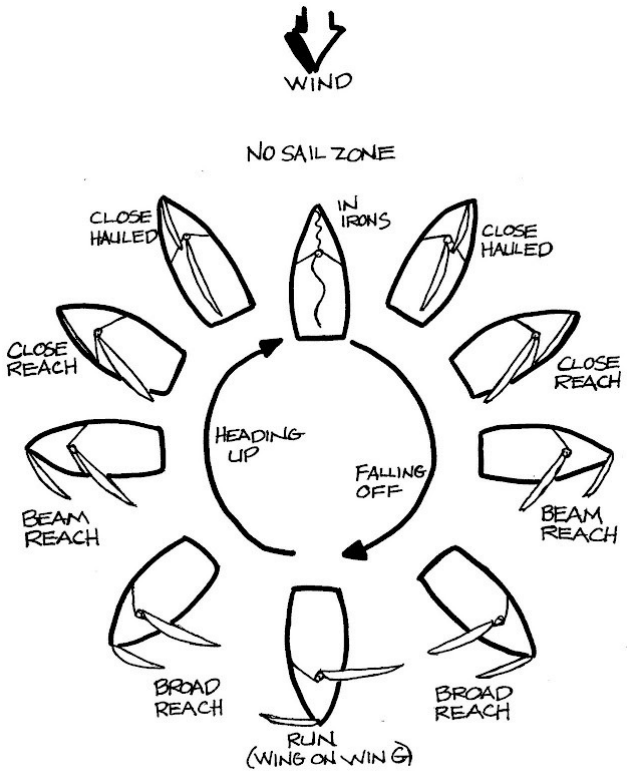

Figure 4. Points of Sail for a sailboat.

Wind controls everything and, as such, a sailor needs to rely on it for everything, including orientation and direction. Wind has to flow over a sail in order to create a force. This force, plus the resistance of the board creates the forward motion. Steering affects the amounts of the forces and so

changes the speed based on where the sailor has steered. In most forms of transportation the force propelling the vehicle is contained within the vehicle, so steering can be in any direction without change in speed. In sailing, an external force is propelling the vehicle. So steering is limited because of the foil of the sail and the physics of the wind. Wind must hit the sail from the side in order to create the force needed for movement If the wind hits the sail straight on there is no pressure on the sail and no force is created. This also means that when the wind hits the sail at different angles between the side and straight on, it will create different amounts of force to propel the board. This process of wind and sail physics is points of sail. The "points" are the common angles at which the wind hits the sail. These angles create different amounts of speed that a sailor needs to understand, both for racing and to get back to the land where he or she started.

This concept of wind applies to all sailboats equally. Windsurfing adds an extra level of complication to the matter. Steering on any kind of sailboat is handled by the tiller and rudder at the rear of the boat. Windsurfing has no such accessory. To steer a windsurfer (sometimes referred to as a sailboard) the sailor must manipulate the sail itself. The sail on a windsurfer has extra axes and degrees of motion that a sailboat sail does not. These elements allow the windsurfer greater control over the points of sail but also make the concept more difficult to understand since there are extra variables. In Figure 4 the sail pivots on only one axis. In Figure 5 the sail not only pivots, but also tilts toward the bow and stern (front and back) of the board (Coutts & Dornells, 2001; Jones, 1987).

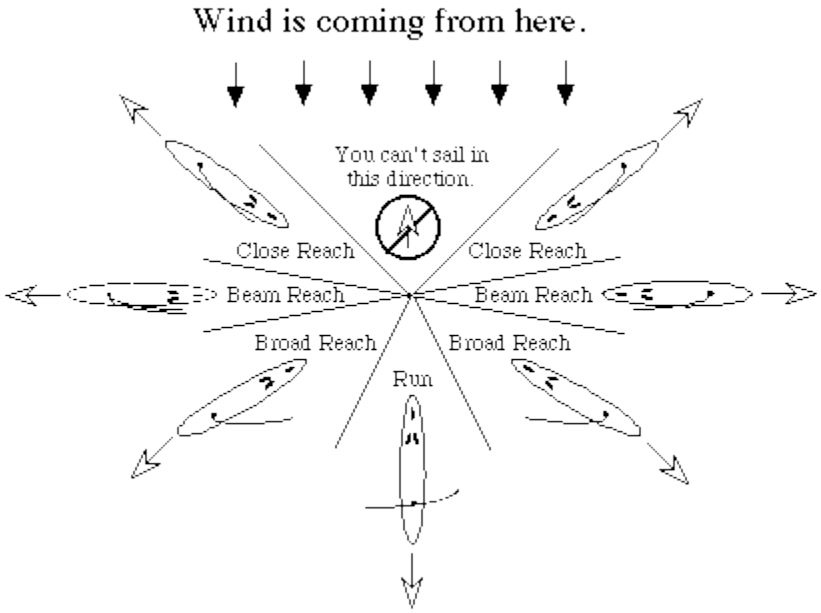

Figure 5. Points of Sail for a windsurfer.

Rationale

Learning from Games

Increasingly more research is being done on the learning that occurs when people play video games (DiGiovanni, 2012). Every aspect of playing a video game teaches something. Everything from abstract ideas to common subjects is unknowingly built into video games. Deciding which quest to

complete first teaches prioritization. Resource management teaches economics. Commanding a squad teaches leadership. Some of these lessons are actively taught, while others are passive (Squire, 2006). Think of this: the use of a compass. Few people have ever had official training in orienteering. However, put a player in a game with a map feature and a compass item, and he will quickly navigate to a location you give him. On top of that, video games have been around so long and have been played by so many people that many of the actions involved in playing games are second nature to gamers.

A gamer no longer has to think about how she is using the map and compass. She is given an objective and starts running while looking at a miniature map with the corner of her eye. She barely even needs a compass since most maps are created with north as "up." She can jump from game to game and automatically learn how to use most of the features of a new game based on its similarity to so many other games. Heads up displays (HUDs) in games are full of information, but players do not stop playing the game to pour over all the information. Players are used to being presented with most HUD features that they can absorb the information, and adapt to new nuances as they play.

Passive learning is something I look to capitalize on in my trainer. By creating a HUD that has sailing elements and other information, but looks and feels like so many other HUDs in so many other games, I plan to utilize passive learning. Presenting points of sail as a compass-like feature will immediately have the players thinking of the dominant compass needle as north, since that is how games most often set up their compass. North is a constant direction that players can orient themselves to. In this simulator, however, "north" is the wind direction. At first players will just take it as "the constant direction they can orient themselves to," but eventually they will realize that it can change slightly. This will motivate them to think a bit deeper. With instructor dialogue and opportunities for text, perhaps in the form of "quests," the player will learn the deeper aspects of this compass. He or she will figure out that the "north" is actually the wind, and as a sailor the wind is what he or she should always orient with. Common features of video games can be used to promote passive learning in this way. Subsequently, they are used to guide a player into active learning once he or she has been given an introduction to a new concept from a passive experience.

A Desire to Grow the Sport

Windsurfing is a sport that falls into many categories. It is family friendly but also an extreme sport. There are no age, size, or physical ability limits to the sport. It is far more accessible than sailing a boat. Yet unfortunately many people are scared off by its seemingly complex nature. Windsurfing is a niche sport in America due to this misconception. In the constant effort to spread the sport and make it more accessible to people there has been little done in the field of instruction. Current instructional methods are certainly adequate and thorough in their presentation and ability to teach, but they are limited by their analogue delivery. Students must either sit on land and read a book, or travel to a location with an experienced professional during ideal conditions. A book can only go so far in a physical sport, and location and conditions may never be ideal or possible in certain areas. Add the need for a trained professional and there is quite a limited supply of learning opportunities. A virtual trainer would both bridge the gap between these two instruction methods, and be infinitely available to anyone, anywhere, at any time. A landlocked person in the mountains could process more information than a simple book can deliver, and know everything except the feel of power in the sail, before heading off to vacation at a sailing resort.

All current instructional material is also designed around the standard status quo of human ability and anatomy. What would happen if a disabled person wanted to learn how to sail? How would

a wounded veteran, a person born without a limb, or someone with a mental illness learn to windsurf? The current instructional methods are static and not easily adapted to unique situations. I have taught several students that fall into unique categories, and I myself am a unique case. I have spent more time than the average sailor would in learning many aspects of this sport. I know firsthand the weaknesses present in the current materials. A virtual learning space would allow ultimate control on the part of the instructor. This kind of control would allow for much easier adaptation to unusual circumstances. Simulations could be run using virtual students to develop methods of teaching sailing to unexpected types of students. Further study could be done on ways this sport, or this method of teaching, would benefit all types of people (Halton, 2008). It is time this sport was brought past its simple days of being a niche market. It should be taken to a new level of accessibility and even further to benefit athletes and users of all kinds (Bainbridge, 2007).

Design Narrative and Results

What Went Right

Holding on to ideas and ideals. It has been a personal dream of mine to merge my two interests, windsurfing and computer art, in some way. Various forms of some sort of windsurfing tool have taken form in my mind and been mulled over. Ever since my undergraduate degree I have been wondering how to successfully bridge the gap between a sport and digital media. I have though of using motion-capture to study windsurfing maneuvers, or creating 3D models of equipment to better inform people before the purchase. I even had an idea to create some sort of tuning simulator that would help users narrow down the optimal tuning settings of sails before rigging them in real life and hitting the water. Through various academic and employment opportunities I kept these ideas alive until I had an opportunity to really develop them. The lesson here is to keep your ideas around, no matter how far fetched they are. At some point the chance will arise that will let you follow through on them. My opportunity came with this project and I certainly was not going to pass it up.

I first learned of the acronym K.I.S.S. doing research on the US Navy SEALS, and it stands for "keep it simple stupid." That was my motto throughout this project and it helped tremendously. I have seen too many people and too many good ideas crash and burn because they bit off more than they could chew. I am a firm believer that delivering 100% of a simpler product is better than delivering 90% of a fancier product.

I knew from the outset what I wanted to create: a virtual windsurfing trainer. However, saying that encompass a very long list of possible topics. So from the start I knew I would be cutting back severely and often. Initially I wanted to do an entire beginner lesson for the sport, from uphauling to stance to sailing. I quickly realized this would be a massive undertaking and began making cuts. The end result, and project proposal, was to teach one part of a beginner lesson, the points of sail. Even that turned out to be a bigger target than I initially imagined, but my dogged adherence to K.I.S.S. helped me to make the cuts necessary to boil things down to an achievable level. In the end it paid off tremendously. I achieved my goal of teaching points of sail in a virtual environment, but due to the simplicity I executed it with there is plenty of room to grow in future development. This was an unintentional benefit. The goal has been reached and the path into the future has already been paved.

A dynamic design process. There are many forms of the video game "production process" used today, and many opinions on which is best. Most commonly known are agile development and waterfall development. Descriptions of both can be subjective, but I will summarize each to give a

better understanding of my own method. Agile development lays out small objective to a larger goal and works on each objective individually, setting completed objectives aside to work on the next (Figure 6). Waterfall development is a straightforward linear process that goes from idea to delivery in one motion (Figure 7). There is no "best" solution to production. Agile development requires much backtracking and constantly testing to tweak small elements. Waterfall development often leads to an ever-growing load of work at the end of the process. With neither of these specifically fitting my project I took to creating my own production process, modifying the waterfall approach heavily. Since I had a clear end goal already set forth I thought it would be best to power straight for that goal. I knew what I wanted and I knew I could get there, much like a waterfall wants to go down the mountain and surely will end up there one way or another. However, I had many unknowns in my production and I did not want to be cornered with a heavy workload at the end of the project.

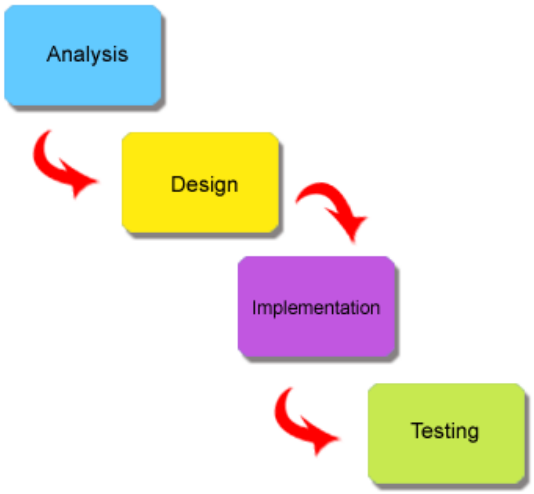

Figure 6. Waterfall Development

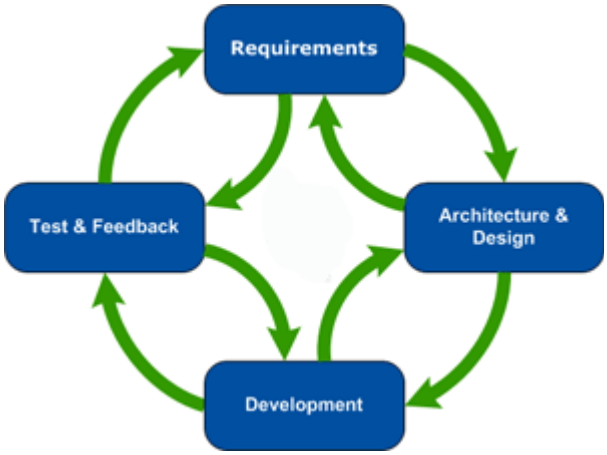

Figure 7. Agile Development

Knowing my end goal but unsure of how to get there I took to front loading my production. I started with what I knew would work without a shadow of a doubt work. This meant working with as few variable as possible, and thus meant working in CE3 with nothing but CE3 tools. A key component of reducing variable was limiting my workspace and setting solid boundaries for myself. In the very beginning I laid out a very small area in and restricted myself to creating this entire tool in that one space. The first step was starting with what I knew was simple, creating terrain in CE3. The second step was quickly laying out clear boundaries for the project. This was done by setting aside a very small area in the level where the player would start. Users will notice that they start on a shore with two shop buildings behind them and the ocean in front of them. I intentionally only allowed myself the small area where the two shops are located as a workspace. This kept me from setting myself up with a content workload that I could not handle.

To demonstrate my adaptive-waterfall development what follows is a before-and-after outline of my production:

My production outline on day one:

1.0: environment
- 1.1: sculpt terrain
- 1.2: texture terrain

2.0: asset creation – environment / objects – shop, channel markers, bulletin board
    2.1: model objects
    2.2: texture objects
    2.3: import to CE3
    2.4: place in environment
3.0: model windsurfer
    3.1: texture windsurfer
    3.2: import to CE3
    3.3: make vehicle entity from windsurfer model
4.0: create wind script in Lua
    4.1: test wind script on windsurfer in CE3
5.0: create HUD in Adobe Flash
    5.1: import HUD into CE3

I was vague on the particulars I had planned. Items like how many objects I would model were left out. Items were also grouped simply and logically: make the world, make stuff for the world, make the main item, make the main item work, call it quits. This was a grand setup for a backlog of work with the traditional waterfall development. There were too many unkowns that would create more work for me later in the production. I knew that would crash and burn and that a dynamic approach had to be taken. I had to be able to address problems early to prevent too much work piling up later.

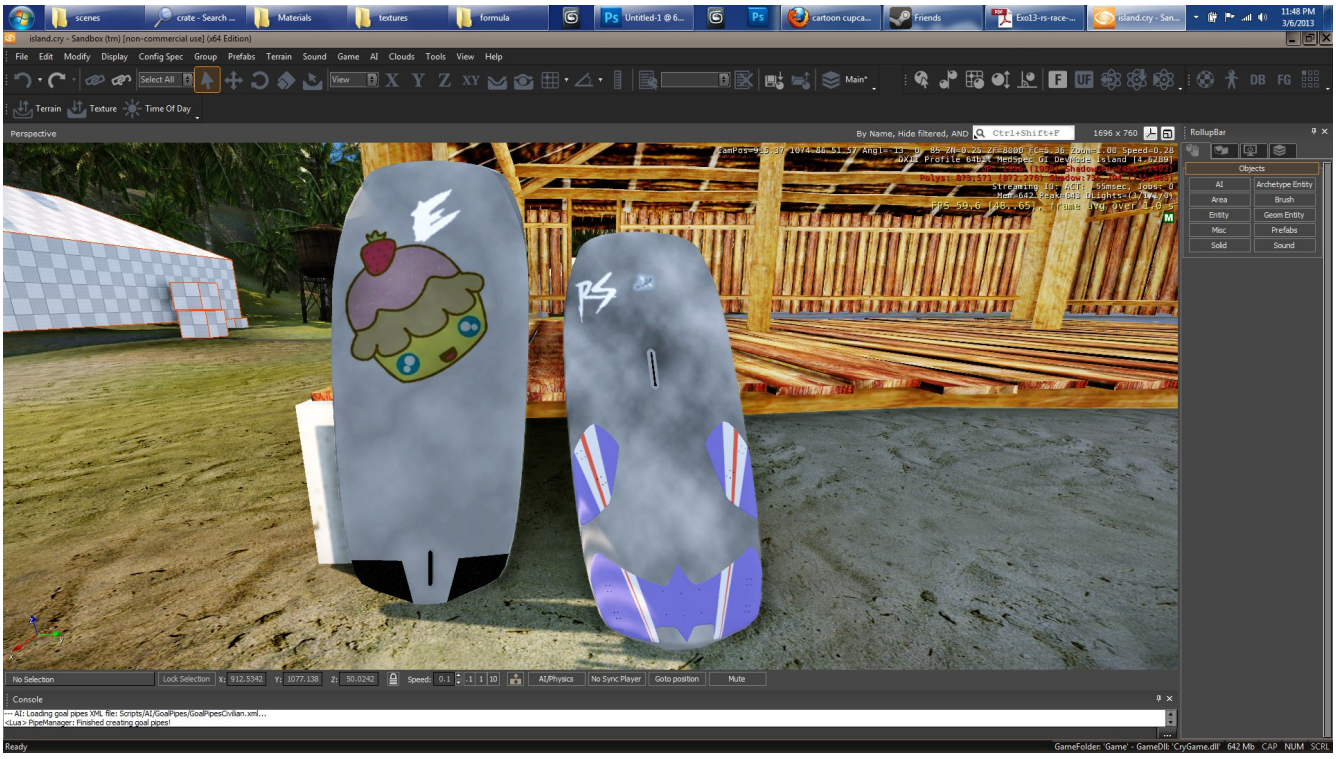

Figure 8. An early version of the formula sailboard.

My production outline once the project was complete:

1.0: environment
    1.1: reference images / terrain research (real-world location)
    1.2: heighmap creation
    1.3: import heightmap to CE3
    1.4: set water level
    1.5: texture terrain from reference images
    1.6: placed vegetation (from CE3 assets)
2.0: asset creation – environment – shop
    2.1: reference research
    2.2: place in environment via "whiteboxing"
    2.3: modeling
    2.4: import to CE3 to test collisions and replace whitebox
    2.5: texture
3.0: asset creation – objects – windsurfing board (sailboard): formula
    3.1: reference research
    3.2: modeling
    3.3: import to CE3 for collision and physics test
    3.4: texture
4.0.0: CE3 scripting and vehicles – wind and a sailboard
    4.1.1: outline script logic with team
    4.1.2: create test environment with default CE3 assets
    4.1.3: create flowgraph script in CE3
    4.1.4: test and debug
    4.2.1: import custom car mesh to replace CE3 assets
    4.2.2: import custom formula mesh to replace CE3 boat assets
    4.2.3: test and debug sailboard vehicle
5.0: asset creation – objects – bulletin board, channel markers, sailboard: kona, sails, buoys
    5.1: model bulletin board
    5.2: texture
    5.3: import to CE3 for collision test
    5.4: channel marker model / import
    5.5: create multiple textures
    5.6: kona reference research
    5.7: model / texture / import
    5.8: sail reference / model / import
    5.9: create multiple textures
    5.10: buoys model / texture / import
6.0: asset placement / world creation
    6.1: test physics on all dynamic objects – sailboards, sails, buoys, CE3 boats
    6.2: fill the shop model with dynamic sailboards and sails
    6.3: create mooring field with dynamic buoys and moored a few CE3 boats
    6.4: set up channel markers, no-wake zones, and shallow areas
    6.5: create a "start" point with bulletin board on a CE3 asset dock
    6.6: set time-of-day parameters
7.0: final sailboard vehicle
    7.1: merge the sail model and kona model
    7.2: import new model into CE3 as a custom vehicle in the main environment

7.3: test and debug kona vehicle
7.4: load wind script into kona vehicle
7.5: test and debug script

There are a couple things of note here. Obviously the list got longer, but it is important to note why and the impact of why. Refer to 2.0 and notice that I only created one asset, and that it was the largest and most complex asset. This gave me a reading on the longest time bracket I would possibly need for asset creation. Next was the formula sailboard (Figure 8). Although less complex than the shop, it was more detailed. This helped me gauge how long it would take to create such details and if they were necessary, which I learned they were not. This initial front loading of the asset production helped me to shift my production around. Instead of continuing on with assets I could put them aside, knowing how long the would take to create, and focus on other big issues. 4.0.0 is broken into two parts, 4.1.0 and 4.2.0. This is because both milestones were tackled at once. My team and I worked together on both scripting and custom vehicles. Both were extremely complex and would most likely have floundered had it not been for the collaboration. Again, I was front loading the big issues to judge the impact of later, similar, smaller issues.

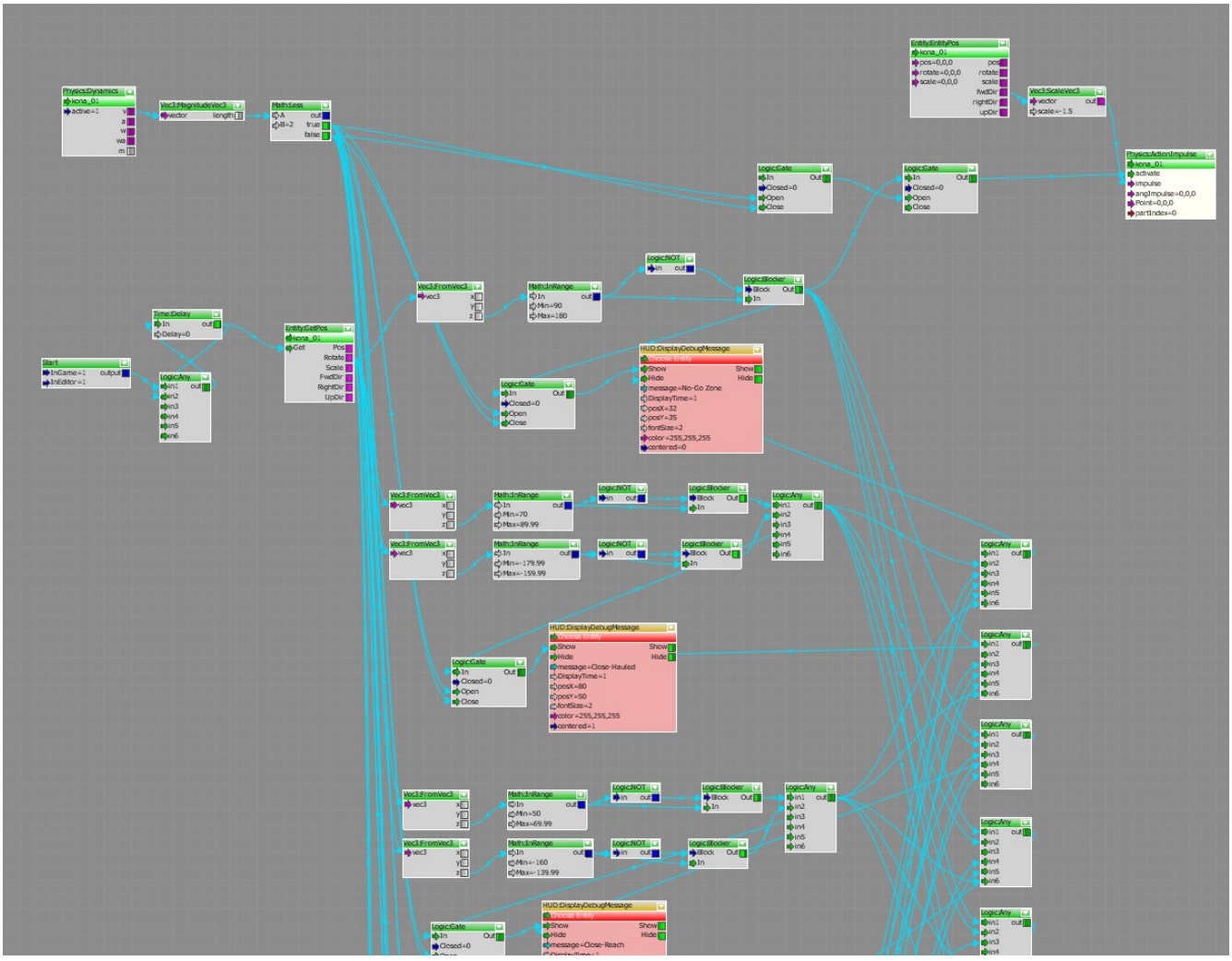

Figure 9. Flowgraph script that controls the upwind speed.

The scripting (Figure 9) and custom vehicle work was all done in a separate environment made specifically for the purpose of testing. This let me make any mistakes I wanted to, some irreversible, without fear of damaging the final product or losing production time. Once these issues were handled the rest of the work could be finished. Initially the scripting was set to be done last, but due to the heavy front loading of the asset creation there was ample time to tackle the scripting and custom vehicle earlier in the process and avoid any last-minute breakdowns in the project.

Also note that my initial work paved the way for better work flow in the end. The later asset creation was done very quickly compared to the earlier method of step-by-step production. Again, the front loading allowed me to gauge what would be needed on simpler objects, as well as practice the techniques for creating them. What initially was a two day process to create a single shop or formula board, became a two day process to create several entities in one sitting.

Making my own rules. What started out as a production outline that looked like a waterfall development process, turned into something much more dynamic. The overall idea of a waterfall, plunging toward an inevitable final goal, was still there. However, this was a bit more of a reverse waterfall, with several streams. Instead of starting out with a small flow and becoming large over time, I started with a large flow and made it shrink over time. It's as if I took a waterfall and dammed or diverted the water that I didn't need. What could have been a rushing torrent was reduced to precisely what I needed. The main brunt of the flow was in the beginning, but each large portion of flow would be evaluated and the resulting evaluation would affect the flow of the remaining process. Anything not crucial to the end result was cast aside, reducing its flow significantly. I took on the assets, found out how much production time the biggest of them would take, then altered the whole production schedule based on that. I altered it so much that I switched topics and began working on another topic. Later on I went back to the assets and picked it up right where I left off, as if I had been plowing along in that same topic the entire time, and kept running with it. I never had to backtrack in my work flow. I answered each unknown up front and left it in a place where it could be resumed later without change of pace. This let me jump from topic to topic while continually pushing forward with my production.

This ended up beautifully complementing my team dynamic as well. I could bring in my team for the scripting and custom vehicles without needing to “bring them up to speed.” Each part started separately and didn't come together until we hard ironed out all the details and the work flow had gone from a torrent down to a trickle. My main scripter and I met initially, brainstormed, then split to work on our parts independently. I assigned separate, but complimentary, jobs to each of us to minimize redundancy. When we would meet we would look over each others' work and continue forward, until at the final step we combined our efforts into the finished product.

This “jumping waterfall” development worked so well that in the end that I was able to create a much more complex environment than I had initially planned. Early designs had channel markers and land as the only obstacles to the player. I had enough availability in my production at the end that I was able to create a significant mooring field for the player to navigate while sailing (Figure 10).

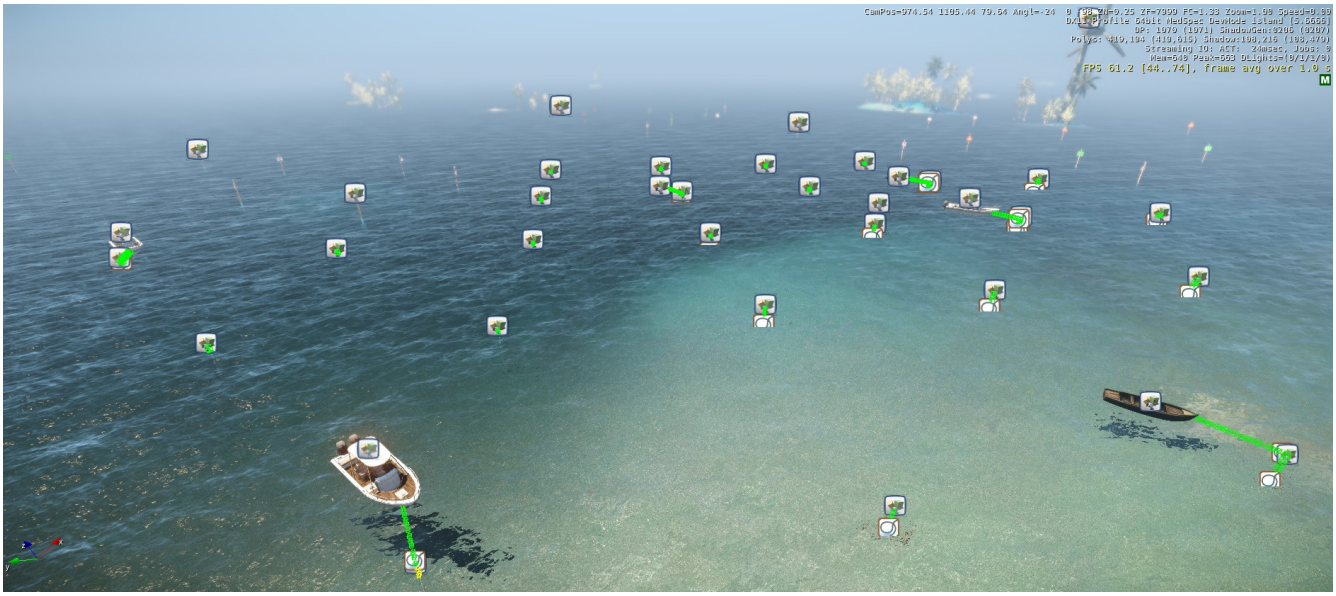

Figure 10. The mooring field setup and channels marked.

What Went Wrong

Where to Turn to for Help. The game engines I have worked with have had varying amounts of support and documentation. Valve's Hammer editor has detailed documentation in wiki format and a thriving community to support it. Bohemia Interactive's VBS engine is on the other end of the spectrum with very few sources of information. Crytek's CryENGINE3 lies a bit in between those two, but more on the Hammer side of the spectrum. CE3 has an excellent online manual, and its previous versions still have strong user bases. Unfortunately CE3 specifically is still rather new and some aspects of its documentation is somewhat foggy. First, there is the free SDK. This product is commonly confused with the mod SDK. The free SDK is for creating fully original content and users are not legally allowed to use any assets from any Crytek product except those included in the free SDK or created by the user. The mod SDK is for creating modifications to Crytek products that are currently on the market, like Crysis 2. I feel that this has unfortunately discouraged most of the masses and as such they have remained in CE2 creating content for gaming. The CE3 developer forums are not very populated these days, most posts are from 2010.

This led to a serious predicament. When I came up against unknown issues in CE3 I had two options. I could sift through the internet to find a solution, or figure it out on my own. Both were fairly decent solutions but both had significant drawbacks. The internet sources were limited to user-made youtube videos or vaguely written forum posts. The youtube videos were amateurish at best, and an instant headache at worst. They were rarely, if ever, a solution to my problems. The forums were not much better as far as quality. I myself made a few forums posts requesting assistance and found it hard to describe specifically what my issue was without writing a novel or play-by-play recounts and uploading my entire project. Searching the forum database for related issues was no better as there were so many topics that were either unrelated or just scratching the surface of what I was dealing with.

The online documentation was a valiant effort at offering help, but it only explained what a tool was, or the basic definition of a particular input field. It left the application or use of the tool up to the user to discover. Combining elements, or a "how to" was never mentioned in the documents.

The flip side to the user-created documentation was solving the problem on my own. This was helped slightly by an included demo "level" with CE3 that had almost everything one could think of implemented. The problem was, there was no manual or explanation. So I either had to take a trial-by-error approach with my own tinkering, or attempt to reverse-engineer a solution from the demo. I spent the better part of three days looking at the included car (a high mobility multipurpose wheel vehicle, or HMMWV) model in CE3's vehicle editor and in 3D Studio Max in an attempt to figure out how to create a custom car vehicle. This was after I had attempted to follow the online tutorials from other users on that very topic and wasted two days.

There were a few printed books on CE3. Sadly, though, they were introductory at best. Two books came from the same publisher and shared many of the same tutorials. In the case of the custom car vehicle the books left me falling painfully short of a working car with, "now press F to enter the vehicle and you can drive around." After the three days I spent reverse-engineering the HMMWV I can testify to how sadly short that tutorial was on actually creating a car. The printed material in general was only enough to leave you stumped. As introductory material and tutorials they were excellent. They taught me how to sculpt the terrain, place trees, and manipulate object in CE3. However as soon as I moved on to more advanced topics they began to falter. Importing original assets from third party programs was covered, but only barely. I could import a model, but why was the collision mesh not aligned correctly? Where was the collision mesh? Why weren't the physics working properly? How do I change the texture? I spent easily more time in the test environment than I did in my final product, and that was with a teammate sharing some of the workload at one point.

Troubles Setting Sail. The most unique part of this project was also its most difficult element. I was trying to create a sailing craft that at least faked the sense of sailing enough to make it obviously different from a motorboat. The problem was that no one, absolutely no one, had attempted anything with a sailing vessel. I found only one person who had even dabbled in watercraft in CE3. Any mention of custom vehicles was about tanks or the secretly buried hovercraft and vertical take-off/landing vehicle code. Even wind was a barely mentioned topic in the forums. Even at this point, with the project in a deliverable state I do not know why there is an environmental wind feature that affects the entire game, and a windarea place-able entity that creates local wind within its boundaries. On top of all this, I could not reverse-engineer a boat vehicle because the boat included in CE3 did not come with a 3DS Max file that users could experiment with. I spent an awful lot of time messing around with the HMMWV and making a custom car vehicle when this project is about sailing watercraft. I had to reverse-engineer the HMMWV to understand how vehicles in general were set up in 3DS Max and CE3, then launch out on my own in an attempt to make a custom boat from scratch. Even though CE3 has a boat vehicle, I could not use it for reference.

I did manage to contact the one user who had done anything even remotely related to custom boats. Sadly that idea quickly revealed itself to be a dead end. Although he had made a tall ship (an old fashioned sailboat) he revealed to me that is was simply a sailboat mesh placed over the speedboat hull. So his sailboat didn't actually sail. Instead it motored around as if it had an engine and the sails flapped with the movement. That was not nearly enough of a fake "sailing" experience to cut it for this kind of teaching too. I needed tried and true reduction in speed when the vehicle turned into the wind.

The Bleeding Edge of Technology. Being on the cusp of technology is great for pushing the envelope, but it has serious drawbacks. Bugs are a big issue. When I first started working on this

project there was an error with the vehicle editor in CE3. It would not open the included HMMWV asset without crashing. So initially I had no choice but to make a stab at creating a custom vehicle through trial-and-error, which is never an efficient process in the sense of a time-restricted production.

To their credit, Crytek listens to its users and makes efforts to fix significant bugs quickly. However, some bugs are more significant than others. Initially CE3 was only set up for Microsoft's Direct X 9. Right before my project began they released a version that was updated for Direct X 11. This was wonderful! The water graphics had shot through the roof and the lighting was so incredibly photo-realistic. Except that it had an issue with one of my video cards and would not open at all. All work hard to be put on hold until Crytek released a fixed version that would at least let me run the engine.

Other bugs, like registry issues, could be fixed by workarounds found by the community, but random crashes were still commonplace with the early releases of the Direct X 9 version of CE3. Apart from that though, there is the issue that this is a tool that is still being developed. Part of the reason for releasing this version of CE3 to the public was to get feedback and input. The application is not fully complete yet, so some features are missing. Significant to my project was the "Save As" feature. Even the books I purchased specifically mentioned not to use the "Save As," because it would only create a new .cry file within your current level's directory. It would not create an entire new directory, with source code and assets, under the new name. As a result, the "Save As" feature had no utility for the project. This made backing up and restoring my project a tedious and manual task. Since there was minimal asset management included in CE3 I had to back up the entire CE3 folder every time. This has made for a very large collection of backups on my external hardrive.

Conclusion

Does it Work?

I believe this is a very successful and productive tool for teaching a very abstract concept. The prototype productive in this project serves as the robust foundation for the development of a strong teaching tool. This is the scaffolding that will be built upon in the future. In this current state the simulation requires a human instructor to be present during the instruction. This blended instructional format is a benefit because it allows the instructor to help the user translate the virtual experience into real world knowledge. However, this instructional format is a downside as well. It means that the tool is not completely stand-alone in its present form. This format is not necessarily a bad thing, but a consideration for future development and application. This tool is an excellent entry into the possibility of virtual instruction for sailing. An extremely realistic and believable environment has been created that is fully interactive. It has been influenced by several real world locations and is a situation that the user could easily find him or herself in.

The elements of a wind-driven sport are here, including indicators for the unseen force of the wind. The physics of the environment are such that flags flutter with the wind, objects on the water are blown by the wind, trees sway, and the ripples of the waves are accurate. Buoys and boats that are tied to something (such as their moorings) will drift and rotate appropriately as they are pushed by the wind's force. All the elements that a user would see in the real world are in place and functioning in my virtual trainer. A user can even approach a moored sailboard and push it around while it floats on the water. The sailboard can be pushed completely into the wind and then the user can step back and watch how the board reacts to the wind's force as it slowly is blown away from the wind until it reaches the end of its tether on the mooring. If the user stops sailing at any point he or she will begin to drift on

the water with the force of the wind. These examples show how the open-ended nature of a game-based simulation allow for the same kind of experimentation that I have used in my real-life windsurfing instruction.

User feedback reinforced the choice to use CE3 for its high level of photo-realism. The environment made the user want to explore and thus use the windsurfer more. With the realistic reaction of the board and rig to the wind the user is forced to sail correctly. Through force of action the user gradually starts to understand how the points of sail operate, even if he or she does not fully realize that. Some instruction must be left unexplained, such as tacking and jibing (turning around), but those topics are not the goal of this tool so that is acceptable. This tool focuses singly on points of sail, which apply to one direction at a time, so users were accepting of the need for a little imagination when turning around. Getting lost in the beauty of the world is part of what windsurfing is all about. There is a point for every sailor where he or she no longer has to pay attention to the technical ability of sailing and just enjoys the ride and the view. Being a virtual world there are no sensory stimulants like feel or smell, so the beauty of the environment and the virtual physical use of the environment have to be higher. With a highly detailed environment, a mooring field to navigate, and many channel markers to dodge I have accomplished a high level of sensory stimulation that had my users engrossed in this virtual world. This prototype serves as the foundation for an effective and realistic tool.

Modifying the Proposed Design

The most notable change from the design proposal is the lack of a custom HUD for the user to refer to while learning. With pushing the K.I.S.S. doctrine it became apparent that a custom HUD was not needed at this time. With the current “scaffolding” setup of this tool it was thought better to go for a more realistic approach. This meant a minimalist HUD with no minimap. The compass feature was left on to mimic most human's ability to remember a compass direction if someone has pointed it out to them on the beach. It's easy to be on a shoreline and have an experienced person tell you “that way is north” and remember it. I left the compass in to account for this, but did not create an interactive diagram of a windsurfer to push the effect of learning this element as one would learn in the real world. However, a slight bridge between the realities of being on one's own and having a magical HUD in front of your face in the form of a text pop-up telling the user what point of sail her or she is on. This is minimal enough to go unnoticed for the most part, thus leaving the user to realistically determine his or her point of sail based on the compass heading.

Less notable is the lack of the user's hands on the boom or feet on the board, and the fixed position of the sail. Both were features that had to be dropped due to the current technical limitations of CE3. In real world windsurfing the sailor must switch sides of the board every time he or she turns around. This means the sailor is technically “driving” from two positions. In the terms of CE3 as a game engine, there is only one allowable “driver” position for a vehicle. Therefore I had to place the driver in one spot and move the camera from side to side on the board to represent switching sides after a turn. Being able to see yourself as the user on the opposite side of the board would have destroyed the realism of the experience. So the only alternative was to hide the user's model completely while sailing. A future fix will be made for this but the source code will have to be edited and that it not an available feature in the free SDK.

The sail is in a fixed position due to technical limitations as well. Although it is possible to have user-controlled moving parts on a vehicle, it has not yet been revealed to the public. A very small handful of users have succeeded in making custom tank vehicles with rotating turrets, however I was unable to translate their methods to a boat vehicle to rotate (or lean) the sail. They were unfortunately

unreachable for help and the traces they left online were not informative. However, as a bright light to the future, it was discovered that CE3 does correctly calculate the impact of a leaned sail on the rotation of the board. In attempting to create a pivoting sail I made three static versions of the Kona, one with the sail raked back, one with it balanced and neutral, and one with the sail tilted forward. Each version of the Kona reacted to the wind as its real world counterpart would. So, currently, the windsurfer is turned via the left and right function and no sail animation accompanies. Hopefully Crytek releases more information about interactive parts on vehicles soon and this can be remedied, as I feel it takes quite a bit away from the realism of the experience.

Future Development and Next Steps

The first priority in future development is to simulate leaning the sail and the physics of that on the board. What this means is that, once Crytek releases documentation on how to manipulate movable parts on a vehicle, I can create a windsurfer that is steerable by the sail. Left and right will no longer control the board, instead the user will have to lean the sail one direction or another to steer the board. This is, to the letter, how a windsurfer works in real life. It was an intended feature to have this happen as simply an animation that would make the user feel like it was real, but true steering in the trainer would still require left and right keys to be pressed. With the physics of the wind reacting with the sail on the board tested and confirmed like this, the realism value of this virtual trained has skyrocketed. For this sport: to steer an object in an unconventional way, and see its real-time implications, all in a virtual environment, is entering into a new realm of possibilities. This is also just a scaffolding of this tool, and is one tiny lesson in the sport that is windsurfing.

My team and I have already begun creating new scripts for the wind that allows the user (or instructor) to change the wind direction, speed, and other environmental factors like the waves. It is also possible to create a non-player character that can be on the water, sailing, with the user. It is a common practice in real life to demonstrate a technique on the water while under way. Features are also being set up that will allow the user to select his or her equipment. Different boards and different size sails have an impact on windsurfing that is hard to determine as a beginer just learning how to hold onto the gear, much less why a big sail behaves one way while a small sail behaves another.

Looking ever farther into the future we have begun brainstorming about advanced applications for specific purposes. Racing, for instance: Set up a race course in CE3 and learn how to race. The abilities to test out the particulars of rounding a buoy would be invaluable to every racer, beginner or professional. Strategies could be practiced solo, with other sailors in multiplayer, or against NPC sailors. Racing is extremely technical and the smallest successes or failures make huge differences at the finish line. Having a realistic practice tool would be in high demand for every competitor.

Future development will also focus on making a wide range of environments, to simulate different kinds of windsurfing conditions. This project's environment is from a combination of a New Zealand archipelago and a Venezuelan archipelago. The shop is fictional but has a setup inspired by a local shop I frequent as well as some images I have seen of a shop in Boniare. With this tool comes the ability to create any location in exact detail. So if a race is to be held at my local venue, I can create that venue, meter for meter, with this tool and allow for precise practice as well as familiarization with the locale. Every sailor does his or her best to know the water that he or she will be sailing in, for safety and enjoyment. In the future I can create a virtual place that allows this sort of information to be explored beforehand, making for a safe and more enjoyable sailing session.

The Future of Windsurfing Instruction

With the amazing possibilities that this tool brings to light, comes the charge to take it further. A significant portion of the US Windsurfing Instructor course is targeted at teaching the disabled. Personally I do not recognize anyone as disabled as I would technically fall into this category, yet I can sail with the best of them. Creating this virtual training tool has shown me what an instructional device like this can do. If I can sit at a computer and realistically manipulate a sailboard by tilting the sail, as I would in real life, then anyone can do it. This tool has revealed that sliding a mouse back and forth can teach someone what will happen in windsurfing when the sail is tilted back and forth. That is learning to windsurf in an extremely simple form right there. I would be hard pressed to think of anything more user friendly for learning something so abstract. This means that for sailors with mental or physical differences they now have access to the purest and simplest tool for learning this extremely complex sport. With all that complexity out of the way much more time can be spent on the unorthodox application of the sport to the particular sailor.

Great time and effort has to be taken to adapt windsurfing to sailors who do not fit in the usual status quo of windsurfing. They have to learn complex abstract ideas while translating the mechanics into their own unique interpretation. Being able to show up to a real world windsurfing spot already having the knowledge of windsurfing in your head and only having the task of physical adaptation to tackle would be an unspeakably great advantage to sailors. Speaking from personal experience I can attest to the countless extra hours I have put into windsurfing analyzing how I have to sail differently with one hand. This is a sport that is in no way inaccessible to anyone. This tool will be a great benefit in helping future sailors realize this and get on the water.

Dedicated To:

Dad and Mom,
who got me addicted to learning, windsurfing and helping people.

My Wife, Michelle,
who feeds my addiction.

God,
who created the wind and water.

Acknowledgments

I owe my deepest gratitude to all the people who helped make this thesis project possible. To Dr. DeVane, who approved such an unorthodox idea and saw it's potential. To Dr. Barnboutis, who took the brave plunge into an unknown game engine with me and helped encourage me. To Dr. Ifju, who was happy to hike across campus just to share the stoke, give me his windsurfing wisdom, and prove to everyone that our sport is as addicting as I keep claiming. To my team, Kyle Ashley and Robby Ronk. Not only did Kyle play a key role in creating the wind script, but he also found himself so hooked that he is now on board for further development. Robby is my constant go-to for helping my scrambled brain slow down and think intelligently enough to develop proper programming logic. To my beautiful wife, Michelle Steward. She wasn't happy just being a supporting fiance and decided to become my loving wife, all while playing second fiddle to my computer screen for many months and reminding me to eat. To my parents, who constantly to me to be God with skin on and go for it. To my brother, who got me into video games in the first place and constantly makes sure he beats me. To Crytek, for creating such a powerful tool and then giving all that power to the public for free. CryENGINE is an excellent tool and this project would never have risen to the level that it is without the free SDK. I am honored to have such support, I could not have done it without you.

Appendix

Figure 1. A real world location recreated in CryENGINE. Adapted from PCGamer Septermber Issue, 2008. CryENGINE3 Screenshot and photograph. Copyright 2008, Crytek. Reprinted with permission.

Figure 2. An example of CryENGINE's exterior evironments. Reprinted from 'CryENGINE," 2013, http://www.crytek.com/cryengine/cryengine3/overview. Copyright 2013 by Crytek. Reprinted with permission.

Figure 3. A screen shot from one of RTI's military simulations. Reprinted from "Media," 2013, http://realtimeimmersive.com/. Copyright 2013 by RealTime Immersive. Reprinted with permission.

Figure 4. Points of Sail for a sailboat. Reprinted from "the horse's mouth," 2009, http://horsesmouth.typepad.com/hm/2009/04/handout-of-the-day-points-of-sail.html. Copyright 2009 by Typepad.com. Reprinted with permission.

Figure 5. Points of Sail for a windsurfer. Reprinted from from "Sailing Terms," 2009, http://socrates.berkeley.edu/~wprinz/windsurfing/Terms.html. Copyright 1995 by B. Prinzmetal. Reprinted with permission.

Figure 6. Waterfall Development. Reprinted from "Software Development Life Cycle(SDLC) & All models," 2011, http://gkmaurya.blogspot.com/2011/04/software-development-life-cyclesdlc-all.html. Copyright 2011 by Blogspot.com. Reprinted with permission.

Figure 7. Agile Development. Reprinted from "Agile Software Development Process," 2009, http://www.managedmayhem.com/2009/05/06/agile-software-development-process/. Copyright 2009 by Managed Mayhem. Reprinted with permission.

Figure 8. An early version of the formula sailboard. CryENGINE 3 Screenshot. Copyright 2013 C. Steward.

Figure 9. Flowgraph script that controls the upwind speed. CryENGINE 3 Screenshot. Copyright 2013 C. Steward, K Ashley.

Figure 10. The mooring field setup and channels marked. CryENGINE 3 Screenshot. Copyright 2013 C. Steward.

Bibliography

Bainbridge, W. S. (2007). The Scientific Research Potential of Virtual Worlds. *Science 371, 472*. doi: 10.1126/science.1146930

Coutts, J., Dornellas, T. (2001). *Start Windsurfing Right!*. Portsmouth, RI: United States Sailing Association.

Crytek GmbH (2012). CryENGINE ® 3 SDK (Version 3.4.0) [Software]. Available from http://mycryengine.com/

Crytek UK Ltd. (2011). Crytek's Official Developers Community. Retrieved from http://www.crydev.net/

Crytek UK Ltd. (2011). Official CryENGINE 3 Free SDK Documentation. Retrieved from http://freesdk.crydev.net/dashboard.action

DiGiovanni, F. "Virtual Worlds." Gametech Users Conference [Conference]. Orlando, FL. 28 Mar. 2012.

Halton, J. (2008). Virtual rehabilitation with video games: A new frontier for occupational therapy. *Occupational Therapy Now, 10*(1), 12 – 14.

Hill, R. (2012, October). Extra Life. *3D World*, 160, 28 – 32.

Jones, R. (1987). *Windsurfing: Basic and Funboard Techniques*. New York, NY: Harper & Row, Publishers, Inc.

Keith, C. (2010). *Agile Development with SCRUM*. Boston, MA: Pearson Education, Inc.

Rice Jr., General, E. A. "Government Keynote." Gametech Users Conference [Conference]. Orlando, FL. 28 Mar. 2012.

RealTime Immersive (2012). CryENGINE. Retrieved from http://realtimeimmersive.com/cryengine

Schmorrow, Captain, D. "Serious Games." Gametech Users Conference [Conference]. Orlando, FL. 28 Mar. 2012.

Squire, K. (2006). From Content to Context: Videogames as Designed Experience. *Educational Researcher, vol. 35 no. 8*, 19-29. doi: 10.3102/0013189X035008019

Tracy, D., & Tracy, S. (2011). *CryENGINE 3 Cookbook*. Birmingham, B27 6 PA, UK. Packt Publishing Ltd.

Williamson, D., Squire, K., Halverson, R., & Gee, J. P. (2005). Video games and the future of learning. *Phi Delta Kappan*, *87*(2), 104-111.

Biographical Sketch

Cody Evan “Stubbz” Steward was born on October 6, 1985 in Tallahassee, Florida. He was the younger of two boys and lived in Tallahassee until he was 8. From there he and his family moved to Arlington, Massachusetts while his dad attended graduate school. After that he moved to Orlando, Florida and graduated from University High School in 2004. He earned a B.F.A. in Computer Animation from the University of Central Florida (UCF) in 2011.

Cody grew up around windsurfers and computers. Before they were big enough to sail, his mom took he and his brother out to sand bars and oyster beds on the nose of the family's Bic 250 sailboard. His dad is always pushing the envelope of computer science and attended MIT for a Ph D. in Computer Science. Cody became a licensed professional windsurfing instructor in high school. He became very passionate about windsurfing and helping others by teaching them, even while pursuing his interest in computer art at UCF. During his time at UCF he worked for several Department of Defense companies as a 3D Artist making virtual trainers and was searching for a way to bridge his two interests of computers and sailing. Soon after his acceptance as a graduate to the University of Florida (UF) he came up with the idea to create a virtual windsurfing trainer to bring his two interests together.

Cody was born without any fingers on his left hand, but this never seemed to faze him. Instead it gave him a unique outlook on life that led him to become passionate about helping others and being an example of one's ability to overcome any obstacle. He uses windsurfing as an example of being able to overcome obstacles even when the idea seems impossibly difficult. Through his work as a windsurfing instructor he met a beautiful woman who shared his passion for life and helping others. Cody and Michelle (also a UF Masters alum) were married on December 28, 2012. Michelle has been a huge support to Cody, pushing and motivating him to create his work not only for his interests, but for the benefit of others like him who may use his work as a way to overcome obstacles in life.