

A GAME SYSTEM APPROACH FOR TRAINING AND EVALUATION: TWO SIDES OF THE SAME COIN

Claudio Coreixas de Moraes¹
Daniel de Vasconcelos Campos²
Roberto de Beauclair Seixas^{2,3}
Michael Aaron Day¹

¹Naval Postgraduate School – NPS
1 University Circle, Monterey, California 93943, USA
e-mail: ccoreixas@nps.edu

²Computer Graphics Technology Group – TECGRAF
Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio
Rua Marquês de São Vicente 255, Rio de Janeiro, RJ, Brasil 22453-900
e-mail: rbs@tecgraf.puc-rio.br, danielaer@ig.com.br

^{2,3}Institute of Pure and Applied Mathematics – IMPA
Estrada Dona Castorina 110, Rio de Janeiro, RJ, Brasil 22460-320
e-mail:rbs@impa.br

KEYWORDS

Training systems, tracking, serious game, virtual simulation.

Abstract

Imagine yourself trying to interpret a theoretical sequence of actions for a given task from a procedures manual. The task implies doing a mental simulation of the spatial representation of all the theoretical actions. Visualizing complex spatial tasks is hard; practicing them in real life for the first time is even more complicated and many times involves correction via feedback from an instructor. We found that two systems that are under development as research projects that can be matched to improve instruction for cognitively complex tasks. Our objective in this paper is an attempt to combine a command and control system and a game-based ship simulator which will create an architecture for a powerful Live Virtual (LV) simulation system. Using this framework, we can bridge the gap between practical and theoretical instruction by providing instantaneous feedback to the instructor and After Action Review (AAR) to the student.

INTRODUCTION AND GOALS

The instructional cycle involved in a navigation and ship handling learning environment, such as a Naval Academy, requires many steps. The first happens inside a classroom where students make first contact with the

theory behind how to handle a ship. In a typical a Naval Academy class students' ages range from 18 to 22 years old; a group of young minds thirsty for more action than a slide show can provide inside a classroom. It is difficult to convey the principles of a large complex system floating in the water without brakes and tires moving on a imaginary highway in a classroom [7] [2]. In order to make the sometimes boring components of navigation and ship handling training understandable, hands on training is required. Most of the Naval Academies worldwide adopt hands on training aboard small Yard Patrol Boats (commonly known as YP) specifically designed for this type of instruction. They function as floating labs where students can experiment and put into practice the abstract concepts of the physics of a ship, and the procedures and rules of the road to safely move a ship from point A to B. Simulation can fill the gap between classroom instruction and hands-on training. This is exactly the concept that drives the design of Yard Patrol Simulator (YPSim), a game-based training system that is under development at the MOVES Department of the Naval Postgraduate School (NPS, Monterey, CA - USA) as a Master's degree project. On the other hand, the Command and Control Support System (CCSS), another Master's degree project developed at the PUC-Rio (Rio de Janeiro, RJ - Brazil), is a tool designed to provide Command and Control (C2) information in a 2D representation [4]. The CCSS design is based on the OODA loop (Observe-Orient-Decide-Act) [3], and presents a virtual environment rich with information necessary for decision makers act on the observation and orientation steps of the process. The type

of information that CCSS primarily handles is for battlefield situations, where unit positioning, weapon range, communications range, and valuable resources can be monitored. Managing this whole set of data rapidly can be crucial and decisive in the course of actions of a military campaign. The CCSS tool is a facilitator in the process, using computational and graphical techniques. To see the value of combining CCSS with YPSim into a new type of VE training tool, it is necessary to understand instruction from two geographically isolated perspectives. From the YP Boat perspective we have an instance of the CCSS onboard the YP, registering and streaming data from GPS, Radar, AIS, engine, rudder and audio/video. From the onshore classroom perspective we have a group of instructors and/or other students observing the training exercise in a close to real time 3D graphical representation of the YP, its state (engines and rudders), other contacts, and audio/video from the bridge, presented via an instance of YPSim. This architecture can optimize the flow of information feedback from the instructor, allow students to share experiences, and facilitate performance evaluation and AAR during a debriefing phase. Allowing the classroom instructor to view a virtual appraisal in a Live Virtual (LV) simulation is something new. Due to physical limitations of the YP and time constraints involved with being onboard, most of the time the classroom instructor is not present at the hands on training exercises, breaking a valuable link in the learning process. The integration between the two subsystems is possible via a 3G network connection using the commercial cellular infrastructure. The nature of the hands on training missions onboard the YPs allows good 3G coverage (less than 10km from urban coastal areas). The data exchange is handled by UDP packets in a Client/Server connection in a close to real time transmission rate.

This work will represent a completely new dynamic in the instruction of navigation and ship handling, where classroom instructors are able to effectively participate and observe students' application of understanding theory.

RELATED WORKS

The use of Open Source software to develop visual simulation designed for naval training was described by Salvatore [14] and Ernst [5] in two Master's degree theses at NPS. The Live Virtual (LV) simulation concept explored in this work is derived from Noseworthy's [10] understanding of Live Virtual Constructive (LVC) simulation. The core of this paper is almost a continuation of Fruchey's efforts to describe the possibilities of integration between LVC simulators for training [6]. In his work, Fruchey describes a very large scenario for battlefield training simulation where LVC instances would be connected to improve the training capabilities of the exercise. He believed that collecting sensor information

from real components in the simulation would be a valuable tool for instructors' and students' analyses during an exercise. The challenge, described by Fruchey as "the most critical", would be the integration of all the components exchanging the data collected. Our work is a simplified extension of Fruchey's approach.

TASK ANALYSIS

It is necessary to provide an overview of what kinds of tasks are performed during hands on training onboard the YPs. Having this concept cemented in the reader's mind will lead to a better understanding of why the proposed integration is important. In order to achieve this goal, we decided to design a very brief task analysis of a general basic instruction at sea, using the Brazilian Naval Academy's (BNA) framework as a base case.

The YPs are the laboratory where students will experiment and effectively learn how the ship's dynamics work, the forces and responses coming from the engine and rudder, how wind and sea currents will affect ship's behavior, and so on. We can observe an example of this kind of vessel in Figure 1. The common situations trained onboard the YPs at the Brazilian Naval Academy are Navigation, Man Overboard (MOB), Mooring, Anchoring and Replenishment at Sea (RAS). These represent the very basic skills that a young line officers need to be proficient in when graduating from any Naval Academy. The focus of the present work is not on the use of YPSim as a tool for the student, but helping the instructor in visualizing and virtually participating in the hands on training process.



Figure 1: The Brazilian Naval Academy's YP (left) and midshipmen at hands on training onboard (right).

Among the training situations listed above, hereafter called "missions," the restricted waters Navigation is the most interesting. Understanding what kinds of tasks are performed, by both student and instructor, during the Navigation mission will guide the analysis of the proposed system architecture (further detailed). Even though students go onboard as a group with different roles for each one of them (representing a navigation team at a ship's bridge), this paper is more focused on the Conning Officer duty. The Conning Officer role is played by one of the students (usually senior grade) who is responsible for maneuvering the ship. He will issue

commands for the helmsman and lee-helmsman concerning engines and rudder respectively. The major goal of the student during Navigation mission is to move the YP from point A to point B, following a given route and respecting the rules of the road applied to general navigation to avoid collision with other contacts. The Conning Officer (our student) will receive suggestions for course and speed from the navigation team (who are responsible for accurately establishing the YP's position on a nautical chart). He/she then decides to act or not by changing engine and rudder status, reflecting on speed and course changes in order to achieve his/her current sub-goal: reaching the next waypoint in the route or avoiding contacts deemed risky. This process is continually repeated until sub-goals are achieved and eventually the YP reaches the end of the navigation route.



Figure 2: Yard Patrol Simulator

At the other side of the instruction process is the instructor. His/her role during the hands on training is a valuable resource to the student in case of doubt or an emergency situation. The instructor's role is not limited to correction of the student's actions or assistance during an unexpected situation. The instructor also observes the student executing his/her plan for conducting the YP from point A to point B. The expertise of the instructor gives him/her sufficient background to qualitatively judge the performance of the student. After every training mission onboard the YP, there is a rich list of topics (both positive and negative) that could, and should, be explored by instructor in an After Action Review (AAR) debriefing. Some cognitive scientists identify this phase of training as one of the most important in the learning process, when the student is able to explore mistakes made during intense cognitive processing and correct his/her actions in the future [1] [12]. Positive reinforcement coming from the instructor is also present during AAR and serves as an important motivational component. One of the biggest problems for conducting the AAR onboard the YP is the lack of a visualization environment where both instructor and student can review a specific situation early in the mission and observe detailed information, such as a turn that was made a few seconds after the ideal time. The heavy cognitive workload that a student, especially a novice, is subject to during missions makes it even harder to remember and associate exact situations with the reinforcements coming from the instructor. The use of a VE representing the mission scenario during AAR would be

a valuable tool that could help the student's recall, using spatial information for an enhanced debriefing. CCSS is a system that is primarily designed to control the flow of information in a saturated complex environment, such as a battlefield [4]. Using CCSS' functionality aboard the YP, instructors would be able to filter, organize and share training information, addressing some common issues faced by both instructor and student. Both instructor and student actions are following an OODA cycle that could be enhanced using CCSS. Another important aspect of the tasks conducted by the instructor during hands on training is real time feedback. This type of reinforcement is provided by the instructor a short time after or before a given situation. Usually real time feedback is given in situations where safety is a concern and a student's actions lead to negative training.

ARCHITECTURE

This section introduces a detailed description of the YP-Sim and CCSS systems, and the proposed Integration Interface. The operation, capabilities and limitations found in both systems will be discussed, focusing on their use as an instructional tool as an integrated package.

Yard Patrol Simulator

Yard Patrol Simulator (YPSim) is a game-based system that is under development as part of a Master's degree thesis at the Modeling in Virtual Environments and Simulation (MOVES) Department at the Naval Postgraduate School (NPS). The application is designed to provide a 3D Virtual Environment using Delta3D [8] [14] [5] open source game engine with a real time physics model of a Naval Academy Training Ship having multiple scenarios as shown in Figure 2. The software's main use is focused on single-user missions where Naval Academy midshipmen (the primary user population) can practice and experiment navigation and ship handling concepts previously acquired in classroom instruction. YPSim hardware requirements allow it to be used on a regular PC with graphics cards supporting OpenGL 2.1 or above and CPU greater than 2.0 GHz. Users can practice some of the maneuvers they will be tested on during the hands on part of their education. From a regular navigation mission, where you can learn about ship dynamics and rules of the road, to a complex Replenishment at Sea (RAS) type of maneuver, YPSim offers an experimental lab where midshipmen can learn practical concepts at a level between classroom and hands on training. Regular LCD monitors are used to provide visual output to the system while regular keyboards/mice are used as default input devices. Using UDP packets under a Client/Server architecture, YPSim provides multi-player capabilities, allowing students to play missions where two or more units are re-

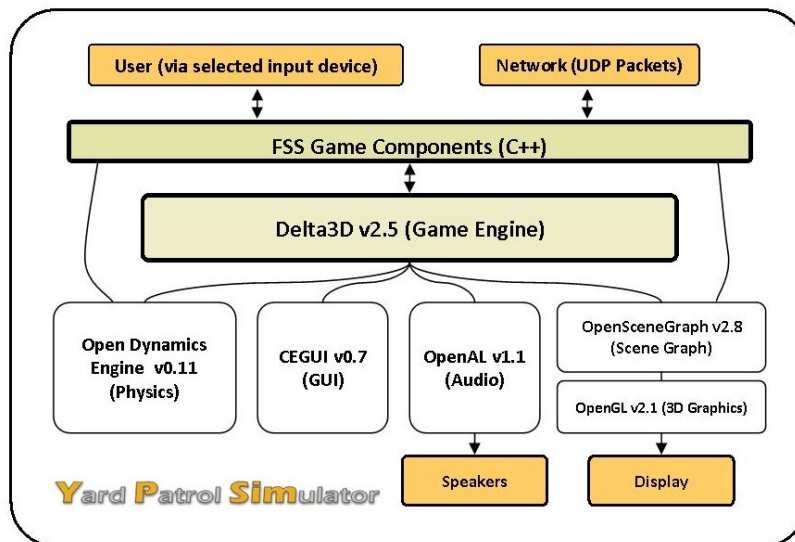


Figure 3: YPSim Overview

quired (like the RAS mission).

Features of YPSim of special interest to an instructor are as a remote 3D scene representation of hands on training. YPSim has a built-in radar module that simulates the YP's navigation radar, being able to receive and plot Radar/AIS (Automatic Identification System) contacts information coming from the real YP. In this way, instructors would be able to share the same contact lists, facilitating the understanding of student's current plan of action. Another valuable source of information is the camera positioning flexibility provided by YPSim, where user can toggle between 4 modes: first-person (inside the bridge), third-person following the ship at a fixed position, third-person orbiting around the ship, and third-person using a fly motion model. Analyzing the scenario from different perspectives is a very important feature regarding AAR in a debriefing phase, when instructors can point out students mistakes from different perspectives. Accurate 3D models of the area of operation can represent position of aids to navigation (buoys, lighthouses, alignments, and land marks) replicating the real world scenario. YPSim uses OS-Geocean to render the ocean surface and affect ship's dynamic response to swell. A situation display constantly presents information about the engines' RPM, rudder angle, heading and depth – providing a good understanding of the ship's status.

Command and Control Support System

The Command and Control theory of John Boyd [3], a 20th century military strategist, allow us to introduce computing techniques that are able to speed up the OODA loop (Observe–Orient–Decide–Act), especially on observing and orienting steps. Command and Control Support System introduces a low cost frame-

work capable of monitoring people, vehicles, boats, or any other elements of interest, almost in real time. The goal of this design is to gather and present the best possible information for decision makers.

For a better understanding about Command and Control Support System, it is essential to explain its pipeline. Looking at the Figure 4 we can identify two main blocks: monitored element(s) and stateful station. The monitored element contains some portable electronics devices such as a small computer, a 3G modem and a GPS logger.

As the proposed framework must be able to monitor from military vehicles or vessels to a soldier, we had to look for hardware that possessed the following characteristics: (a) light, (b) small, and (c) good battery life. So, we found on the market the Igot-U GT-600 GPS logger, the PC2 computer, the Tekkeon MP3450 R2 battery, and the MIMO monitor according to these requirements.

We should note that these characteristics are mainly required by “monitored elements”, as the “stateful station” could be installed in a room. In this case, the devices used need not be light or use external batteries.

Figure 7 shows the hardware used by Monitored Element(s).

The main idea was to write a script, in our case we use Lua [9], able to do the parsing of National Marine Electronics Association (NMEA) data, received by the GPS logger, and transmit/store a record with information such as current speed, latitude, longitude and travel direction. After the parsing we insert all this information in two databases: local (monitored element) and remote (stateful station). The connection with the remote databases is established through a 3G modem. Actually, any remote connection technology can be used (wi-fi, radio, satellite...).

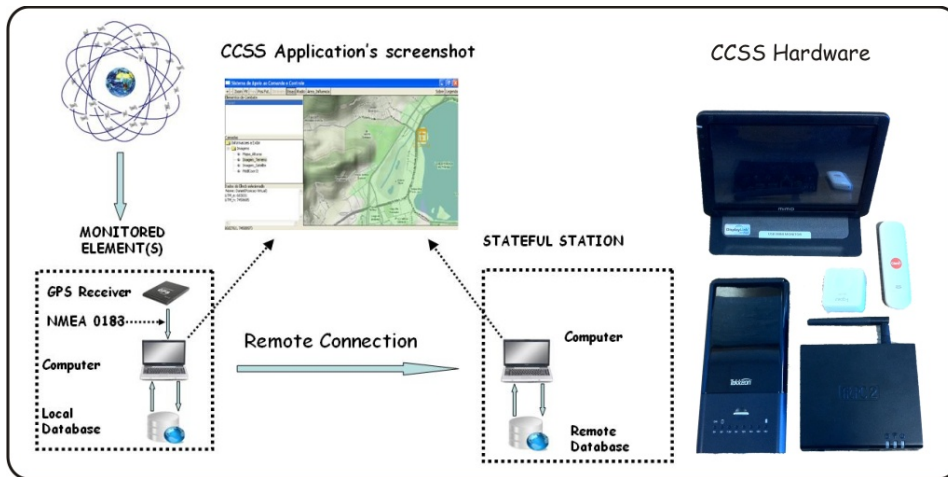


Figure 4: Architecture of the system to support C2

Using a specific application built to handle such data, both members of the system (monitored element and stateful station) can access a display that shows an icon representative of the georeferenced monitored element. The framework is still able to keep audio and video communication between the monitored element and the stateful station through any communication software such as Skype, MSN, Ekiga...

An example use case can be found in [4]. In this case, the application developed has the goal of providing, in a unique environment, some relevant information to the high command, such as:

- 1– Sketch graphically, in interactive time (near real time), the monitored elements of a force deployed in theater operations as well as the information about speed and direction of travel of these elements;
- 2– Provide information graphically, about the line-of-sight, the weapon range and the radio range of the monitored elements;
- 3– Present thematic layers with various coordination and control measures used in military operations, such as goals, boundaries, landing beaches, among others;
- 4– Record in a database positions occupied by monitored elements allowing the reconstruction of their itinerary;
- 5– Enable virtual position simulation of monitored elements, allowing the establishment of retransmission stations.

We can observe a screenshot of this application on Figure 5.

Integration Interface

Despite both systems' (CCSS and YPSim) network capabilities, they are not primarily designed for this in-



Figure 5: Weapon range of 2_CiaCC(in red)

teroperability mode. YPSim and CCSS do not share the same network protocol for message traffic and this is the major issue to be addressed in this work. Another consideration needs to be made regarding both applications' handling new data fields that are required for this integration. In this subsection we will present solutions for a common protocol and what changes need to be made in both applications, such that the integration is possible for this specific use as a training tool.

The first consideration is about the audio and video connection. CCSS relies on freeware VoIP applications such as Skype or MSN to handle this communication between instances [4]. This work is not intended to modify this simple, but efficient solution preserving the same concept to the proposed architecture. Regarding the YP state data package (position, course, speed, engine RPM, rudder angle, etc), CCSS originally used TCP socket connections between its instances for messages exchange. This approach assures reliability in the network but has the drawback of higher latency, a big issue for real time simulations [11]. YPSim, at the other hand, adopts a UDP socket connection using a very simple and fast protocol composed of two types of data packets in a Client/Server connection. It is clear that both systems cannot talk to each other without changes in the

network implementation following protocol standardization. Between the reliability and high latency of TCP and uncertainty and low latency of UDP, we opted to use UDP sockets in our integration. The system relies on a Client/Server architecture with very simple packets. The Server, represented by an instance of YPSim running in a classroom or office, opens UDP port 3000 and waits for a connection. The Client, represented by an instance of CCSS running on board the YP during the training exercise, requests a connection by sending a `CONN_REQUEST` packet containing its ID number. Once the Server acknowledges the request and accepts it by sending a `CONN_ACCEPTED` packet, the Client starts sending data packets containing information for the virtual simulation on YPSim. In order to represent the VE instance of the training exercise that is happening on board the YP, CCSS sends out two types of data packets: `YP_DATA` and `CONTACT_DATA`.

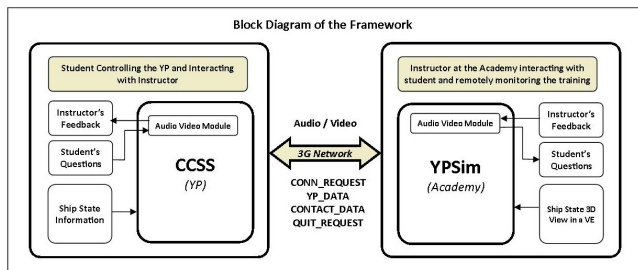


Figure 6: A block diagram representation of the integration

The first packet, `YP_DATA`, holds information about the YP state at a given moment that will make possible the virtual representation of the YP in YPSim. The virtual simulation needs information about what is happening at that moment inside the YP concerning the navigation/ship handling exercise. In order to do that, this packet contains the following fields:

- Header (packet name + YP's international call sign);
- YP's position (LAT/LONG);
- YP's heading (0-359);
- Course (0-359);
- Speed (in knots);
- Turning rate (in degrees/sec.);
- Linear Acceleration (single value for speed);
- Angular Acceleration (single value for heading);
- Engine RPM (two floats, one for Port Engine and one for Stbd Engine);
- Rudder Angle (one float, assuming both rudders acting in sync);
- Apparent Wind (two floats, one for speed and one for direction).

YPSim uses a transmission rate of one data packet every 15 frames, or two packets per second considering an average of 30 frames per second under normal operating conditions. This heartbeat rate [13] was proven to be satisfactory in previous tests of YPSim thanks to the low velocities (linear and angular) involved in the simulation and second order dead reckoning algorithm used (accounting for velocity and acceleration) [13]. Using the information provided by CCSS running on board the YP, the instructor is able to understand and better visualize the YP's location, its movements, and governing actions applied by the student (rudder and engines). The instructor virtually follows the student's actions and qualitatively evaluates if his/her decisions meet the best practices according the theory of ship handling and navigation. The apparent wind information also provides a valuable resource to better understand the environment where the exercise is taking place.

The second packet, `CONTACT_DATA`, holds information about other ships in YP's vicinity. This information can be extracted primarily from an Automatic Identification System (AIS) equipment, when installed on board the YP. Information about contacts close to the YP help the instructor to visualize the scenario and understand student's intentions while maneuvering. This data packet can be less complete and contains only the basic information necessary to make corrections in YPSim, applying only a first level of dead reckoning via velocities (speed and turning rate). The following fields will represent our `CONTACT_DATA` packet:

- Header (packet name + contact's international call sign + type of ship);
- Dimensions (length and beam);
- Position (LAT/LONG);
- Heading (0-359);
- Course (0-359);
- Speed (in knots);
- Turning rate (in degrees/min.).

A secondary source of contact information, for building a `CONTACT_DATA` packet, can be navigation radar equipped with Automatic Radar Plotting Aid (ARPA) capabilities. When using Radar as a data source, the Tracked Target Message (TTM) NMEA sentence is parsed and position is converted from bearing and range to Lat/Long. TTM sentences coming from radars do not provide as detailed information about the target as the AIS does, therefore these fields (call sign, type, dimensions and turning rate) in this package will be filled with 9, flagging radar as the source. Because of the nature of the source of information (AIS or Radar), contacts are not updated as frequently as the YP itself. This implementation will consider a fixed heartbeat of 5 seconds

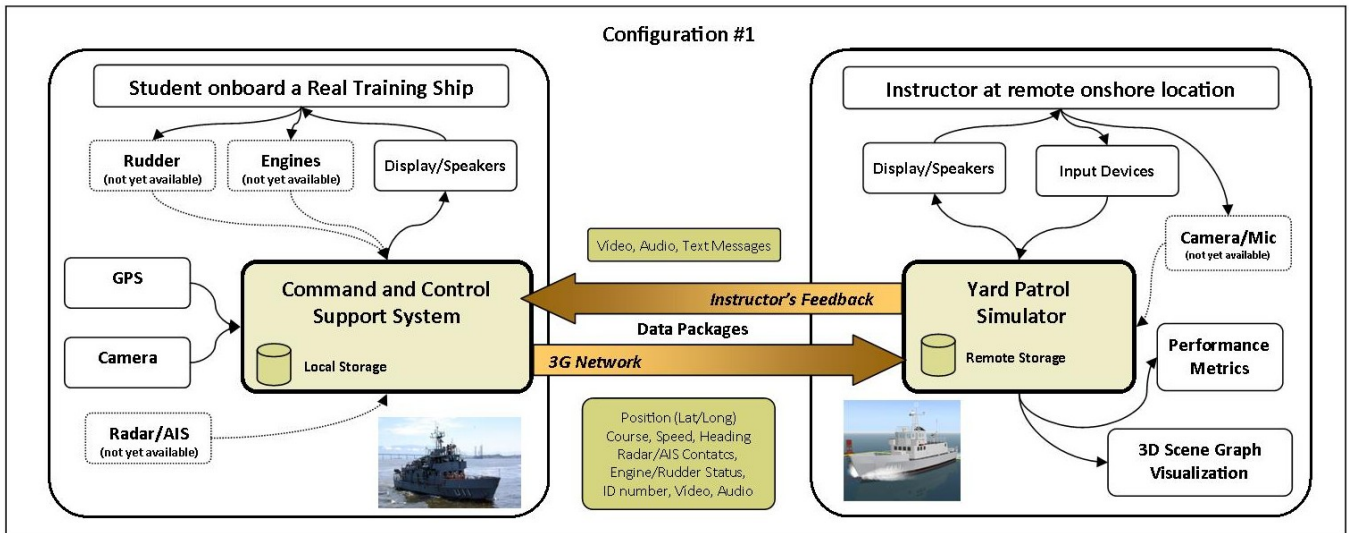


Figure 7: The integration between CCSS (student onboard the YP) and YPSim (instructor remotely onshore)

between data transmissions of the same contact. Position and orientation of contacts between updates are calculated using a first order dead reckoning algorithm that uses only course and speed as parameters [13]. Contacts that are not updated after a timeout of 120 seconds will be automatically removed from the virtual simulation running on YPSim.

Another auxiliary packet is required to remove entities, both YPs or contacts, from the simulation. The QUIT_REQUEST packet is responsible for this job, transmitting to the Server the ID of the entity to be removed, i.e. entity's call sign. Upon receiving this packet, the Server (YPSim) automatically removes the entity from the world without sending any acknowledgement message back. After sending the QUIT_REQUEST packet, the Client automatically stops sending updates relative to that contact or itself. The QUIT_REQUEST is just an attempt to advise the Server that the contact's updates will cease. If the Server does not receive this UDP packet, a timeout will fire and the entity will be removed after 120 seconds of inactivity.

CASE STUDY

A typical application of the proposed framework is for a student that needs to learn how to execute a precision anchoring task. As the Conning Officer of a YP, he/she learned a series of complex procedures to correctly move the YP from a given position A to the anchoring position B. Ship handling manuals could vary a little on how to execute this maneuver, but the precision anchoring task will be composed of basic navigation (as described in Task Analysis Section), with systematic speed reductions as the anchoring position approaches. A set of verbal instructions must be passed to the Boatswain,

located in the YP's bow, reporting readiness for anchoring as the ship approaches the goal position. Suppose a precision anchoring task is executed on board a given YP under local supervision of the YP's Commanding Officer (CO) and remotely monitored by the student's classroom instructor located at the Naval Academy's navigation laboratory (NavLab). One week before, the Naval Academy Instructor taught all the procedures of the precision anchoring task to student and his classmates. The instructor is not confident that his class mastered this topic, since it has some complexity that is hard to memorize from classroom slides and procedure manual pictures. This week students will be tested on this topic during a hands on training exercise on board the CO's YP. The instructor and a group of students will be able to remotely observe this activity by using a version of YPSim that is installed at the academy's NavLab. This instance of YPSim is linked to a version of CCSS running on board the YP, transmitting data for position, engines, rudder, apparent wind and other contacts tracked from Radar or AIS. YPSim and CCSS will also be connected by audio and video streamed using the same link, a 3G commercial cell phone network. The instructor hopes to gather enough information to evaluate how well his instruction was retained from one week before, focusing on improvements for future classes. He also wants to give a chance to highly motivated students from other classes to remotely observe this maneuver, hoping to trigger discussions about the actions taken on board. Even though the YP is 5 Nautical Miles (NM) from the Naval Academy, the instructor can interact with his student and the YP's CO.

Imagine now all of the many actions that occur in a given training exercise on a given day. In a debriefing with the instructor the next day of class, it is unlikely

that a sufficient part of the instruction will be retained. It is true that good and valuable feedback is provided by the YP's Commanding Officer during the hands on training or even minutes after the exercise. The point explored in this framework is the possibility of recording exercise data in a database for later debriefing and a possibility of interaction in real time between the classroom instructor and his/her student, a key link in the instructional process. The instructor has now a chance to (at any later time) evaluate the level of instruction provided in the classroom, check if the methods used fit the hands on training expectations and also qualitatively evaluate how the students transfer the knowledge from the theory to the practice.

CONCLUSIONS

This work presents a framework that integrates two conceptually different systems that could work for the same purpose: improving navigation and ship handling instruction at Naval Academies. Integration between CCSS and YPSim is possible using UDP socket connections via a 3G cell phone network with a few changes both of the existing software infrastructures. An integration interface can be created, making data transfer possible from a YP engaged in a training exercise to the monitoring station at the Naval Academy. It is important to reinforce that the Live Virtual (LV) simulation framework presented in this work is generalizable. In this work we used an existing ship handling and navigation simulation (real YP + YPSim), however environments such as medical, battlefield or flight simulations, could also benefit. Virtual representation of the monitored student during hands on training expands the eyes of the instructor to a new horizon that extends beyond the walls of a classroom, providing a better instructional environment for complex tasks.

FUTURE WORK

The concept proposed in this work needs experimental proof. Higher levels of interoperability with other Live Virtual Constructive (LVC) simulations could be achieved using DIS or HLA protocols, instead of a simple UDP. CCSS is not ready to handle AIS and TTM NMEA sentences, but an extension of the original parsing capabilities of CCSS could be made. There is no reason not to have multiple Client connections with the YPSim Server, although this work was focused on an architecture using a single instance of CCSS connected to YPSim. The use of multiple instances of CCSS and YPSim could be useful in future work exploring tactical maneuvers exercises, where more than one YP is engaged in the same task.

References

- [1] Annet, J. 1989, "Training Skilled Performance. Acquisition and Performance of Cognitive Skills".
- [2] Barber, J. A. 1934, "Naval Shiphandler's Guide".
- [3] J. R. Boyd. *The Essence of Winning and Losing*. 1995.
- [4] Campos, D. V.;Seixas, R.B. Command and Control: A low cost framework to remotely monitor military training. In *Spring Simulation Multiconference - SpringSim '11*, 2011.
- [5] Ernst, R. B. 2006, "The Design of a Stand-Alone Division Tactics Simulator Utilizing Non-Proprietary (Open Source) Media and Iterative Development"
- [6] Fruchey, D.B. 1994 , "Distributed simulation for live training"
- [7] Hooyer, H. H. 1983, "Behavior and Handling of Ships".
- [8] Delta3D 2011, <http://www.delta3d.org>. Retrieved on 2011-04-13.
- [9] R. Ierusalimschy. *Programming in Lua*. Lua.org, 2006.
- [10] Noseworthy, J. R. 2008, "The Test and Training Enabling Architecture (TENA) - Supporting the decentralized development of distributed applications and LVC simulations"
- [11] Sales, L. M., Almeida, H. O., Perkusch, A. 2008, "On the performance of TCP, UDP and DCCP over 802.11g nNetworks"
- [12] Percival, F., Ellington H., Race, P. 1993, "Handbook of educational technology"
- [13] Ryan, P., Oliver W. 2006, "Modelling of Dead Reckoning and Heartbeat Update Mechanisms in Distributed Interactive Simulation"
- [14] Salvatore, R. B. 2005, "Using Open Source Software in Visual Simulation Development"