Visually Realistic Graphical Simulation of Underwater Cable

Ori Ganoni
Department of Computer Science
University of Canterbury
Christchurch, New Zealand
at time *t*
ori.ganoni@pg.canterbury.ac.nz

Ramakrishnan Mukundan
Department of Computer Science
University of Canterbury
Christchurch, New Zealand
mukundan@canterbury.ac.nz

Richard Green
Department of Computer Science
University of Canterbury
Christchurch, New Zealand
richard.green@canterbury.ac.nz

ABSTRACT

This paper presents different modeling considerations that are important in simulating visually realistic behavior of underwater cables attached to remotely operated vehicles. The proposed methodology has been tested on highly complex models of aquatic environments created using Unreal Engine 4. Current methods and implementations of cable simulations that are widely used in computer graphics are generally suited only to light density mediums such as air. In this paper, we present modifications to the above model required for simulating neutrally buoyant cables in underwater environments. The simulation results presented in this paper successfully demonstrate different behavioral aspects of flexible variable length underwater cables and their variations with respect to modeling parameters using our proposed method.

Keywords

Robot simulation, ROV, Underwater simulation, Cable simulation, Unreal Engine 4.

1 INTRODUCTION

Cables are used widely in underwater environments for power supply and communication to remote locations and to support various types of underwater structures. High tension cables are generally used to tow fishing equipment and research probes whereas low tension cables and ropes are used in underwater tethered systems like Remotely Operated Vehicles (ROVs) (Figure 1). The drag introduced by the water medium and the buoyancy forces make the underwater cables behave differently to less dense mediums like air or vacuum. For most ROV cables (and also in our simulation), gravitational effects can be ignored since as a design goal, underwater ROV systems use neutrally buoyant cables. Simulating the behaviour of the cable attached to an ROV can be helpful in several situations such as (i) developing control algorithms for avoiding cable tangling, (ii) handling vision issues when the cable is in front of the ROV's camera, (iii) estimating the configuration of the cable in different conditions and manoeuvres, and (iv) simulating complex manoeuvres for missions involving multiple cables and ROVs.

Figure 1: Visual simulation of Underwater tethered ROV

The behavior of a rope under external forces is generally modelled as a multi-rigid-body dynamic system (or chain system), by representing the rope as long chain of segments. In order to simulate a large number of segments in real-time, we need a fast and memory efficient method which need not necessarily be physically accurate. At the same time, the motion should look physically realistic, and controlled by a number of parameters which defines the cable behaviour. In addition, some of the unique characteristics of the water domain need to be parameterized and added to the simulation model. Simulation of dynamic systems in computer graphics mainly use force-based methods, where linear and rotational accelerations are computed from forces and torques. A time integration method is then used to update the velocities and positions of the object.

In contrast, geometry-based methods work directly on vertex positions, modifying them using iterative update equations. The main advantage of a position-based approach is its controllability. In force-based systems, overshooting problems associated with explicit integration schemes can be avoided. In addition, collision constraints can be handled easily and penetrations can be resolved completely by moving points to valid locations [1]. The position based simulations which were originally developed for the simulation of solids were also extended to the area of fluid dynamics [2] where it is not reasonable to simulate the interacting forces between the particles in real-time. In contrast to force based simulation, position based models can scale up and can be used today to simulate large multi-body systems like fabric or fur in real-time. While force based systems tend to be physically accurate under certain assumptions, position based systems aim to be visually realistic. For example, a set of simulation parameters may not correspond to real physical values, but may generate realistic object behavior as the output. The stiffness parameter implemented in the Unreal Engine which we will later discuss is an example of a parameter which influences the dynamic behaviour of the position based cable model but has no meaningful physical value (like mass or length).

Our proposed simulation method in terms of number of particles lies between the forced based system and the position based system. In simulating a long cable with a large number of segments, we aim for realtime performance by applying constraints and also use position based models. In our underwater rope implementation, we take the existing Verlet integration suggested by Jackobson [3] and modify it to simulate a realistic underwater cable.

1.1 Our Contribution

The existing rope simulation in Unreal Engine performs in a convincing way only in a light density environment like air but looked quite non-realistic in the aquatic medium. This motivated us to return to the original assumptions of the simulated implementation and modify it in order to create visually convincing underwater rope/cable simulation. In our work, we added extensive damping and random displacements to the particles and experimentally analysed and verified the resulting behaviour. Specifically, we were interested in the behaviour of variable length long cables attached to a robot at one end and to a spool in the other end as can be seen in video [4]. That kind of cables have some unique physical characteristics that could not be addressed by the current features in existing simulation frameworks. We demonstrated the behaviour of the rope as part of a larger underwater ROV simulation using Unreal Engine 4.

This paper is organized as follows: Section 2 describes the related work done on underwater cable simulation in the mechanical engineering domain and the position based simulation work done on cables in the computer graphics domain. Section 3 describes in detail the theory of position based methods with focus on cable simulation in addition to our proposed model. Section 4 describes the software we used and developed for the purpose of this work. Section 5 analyzes the results of the cable simulation. Section 6 concludes the paper with a summary of the important concepts and results presented and also outlines future work.

2 RELATED WORK

Most of the underwater cable modelling was done in the mechanical engineering domain using force based methods. Cable and chain models in general are simulated using a segment based model [5]. Buckham [6] used force based methods to simulate a cable model for use in low-tension dynamics simulation. He presented a computationally efficient and novel third-order finite element technique that provides a representation of both the bending and torsional effects and accelerates the convergence of the model at relatively large element sizes. In his paper, he managed to reduce the number of state variables defining the cubic elements of the more conventional finite element approaches . Other water cable related work reported in literature involved towed cable systems. High tension systems like towed system are quite common and a lot of work has been done on that topic. Wang investigated in his paper the parameters influencing the manoeuvre of towed cable system dynamics [7]. Lambert created a model for the dynamics and control of towed underwater vehicle systems [8]. Gonzalez created a simulation of cable pay-out and reel-in with towed fishing gears [9]. Ablow simulated the behaviour of a long cable pulled in a circular pattern [10]. Some work has been done to model the bending and the stiffness of underwater cable systems [11] [12]. Most of the simulation in the mechanical domain were designed to meet specific purpose or requirement and to serve as a guide for cable system design. For the variable length case, Prabhakar [13] developed a dynamic simulation of variable length tether in a tethered underwater vehicle system.

Position based methods are used widely in the computer graphics domain. Jackobson [3] described in detail the position based model for cable simulation. His work was the basis for the current implementation in today's game engines [14]. Our work is based on the survey paper by Bender [1] on different position based methods currently used in computer graphics.

3 ALGORITHM OVERVIEW

The Unreal game engine uses the Verlet integration method for rigid multibody simulation presented by

 ISBN 978-80-86943-41-1

Varlet [15]. The heart of the existing rope simulation is a particle system. each particle has two main variables: Its position x and its velocity v. The new position $x_{t+\Delta_t}$ and velocity $v_{t+\Delta_t}$ are computed by applying the rules:

$$x_{t+\Delta_t} = x_t + v_t\Delta_t \quad (1)$$

$$v_{t+\Delta_t} = v_t + a_t\Delta_t \quad (2)$$

where Δ_t is the time step and a_t is the acceleration. For obtaining a velocity-less representation of the above scheme, instead of storing each particle's position and velocity, we store its current position x and its previous position $x_{t-\Delta_t}$. Keeping the time step Δ_t fixed, the update rule (or integration step) is then:

$$x_{t+\Delta_t} = 2x_t - x_{t-\Delta_t} + a_t\Delta_t^2 \quad (3)$$

$$x_{t-\Delta_t} = x_t \quad (4)$$

$$x_t = x_{t+\Delta_t} \quad (5)$$

Jackobson [3] suggested in his paper that by changing the update rule to $x_{t+\Delta_t} = 1.99x_t - 0.99x_{t-\Delta_t} + a\Delta_t^2$, a small amount of drag can also be introduced to the system. This is a useful equation that can be further modified to add large drag or damping to a system in an aquatic environment. In our implementation, we added small random displacements for creating micro current effects suitable for ocean-like environment. Those micro currents prevent the rope from looking frozen in space where there are no other forces presented to the simulation. This is usually the case in long low tension cable characterized by tethered systems. Our proposed final model can be summarized by the following equations:

$$x_{t+\Delta_t} = x_t + (x_t - x_{t-\Delta_t})D_r + a\Delta_t^2 + r \quad (6)$$

$$x_{t-\Delta_t} = x_t \quad (7)$$

$$x_t = x_{t+\Delta_t} \quad (8)$$

where D_r is the drag coefficient with maximal value of 1 (no drag). It is set to 0.9 to introduce a large amount of drag typical of aquatic systems and is multiplied by the velocity term $(x_t - x_{t-\Delta_t})$ When the time step Δ_t is set equal to 1 for simulation purposes. r is the added random displacements to simulate random forces generated by underwater micro-currents. r was uniformly distributed and limits were chosen to be small enough so the random behaviour will only cause long-term effect on the cable.

The next step of the rope simulation is to apply the distance constraint. This means that the distance between adjacent particles should be kept constant. This process is done iteratively by pushing the particles directly away from each other or by pulling them closer to maintain the required distance. The following pseudo-code (Figure 2) describes this process:

```
SolveDistanceConstraint(PosA,PosB,
    TargetDistance):
 Delta = PosB-PosA
 ErrorFactor=(|Delta| - TargetDistance)/
               |Delta|
 PosA += ErrorFactor/2 * Delta
 PosB -= ErrorFactor/2 * Delta

SolveConstraints():
 for iter=0 to SolverIterations
  for ParticleIndex=0 to NumOfParticles-1
   SolveDistanceConstraint(
    Particles[i],Particle[i+1],TargetDistance
        )
  for ParticleIndex=0 to NumOfParticles-2
   SolveDistanceConstraint(
    Particles[i],Particle[i+2],2*
        TargetDistance)
```

Figure 2: Distance constraint algorithm

This pseudo-code above shows how distant constraints are implemented in Unreal Engine 4. We can see that the "SolveConstraints" function has one outer loop which is responsible to perform iterations to enforce the constraint. More solver iterations will give more stiffness to the cable. In Figure 4, the length of the rope is changed when introducing a force at one of its ends and changes in the overall length is dependent on the number of constraints and iterations applied to the rope particles. The second inner loop reduces the flexibility of the rope by enforcing constraints between particles that are separated by one other particle. That specific constraint limits the ability of the rope to bend. Figure 3 shows the difference between the two constraints. The bending constraints were useful in smoothening out the effects of random displacements added to the underwater simulation. Finally, our final solution was implemented as a new underwater rope plugin.

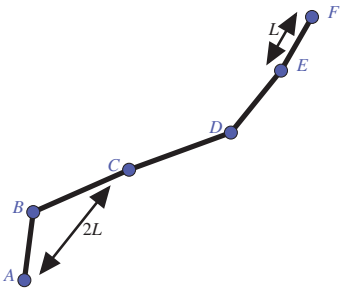

Figure 3: Length constraints and bending constraints. We can see at this diagram an example to the constraints apply on the cable segments. For example between adjacent points like E and F we require L distance which will create resistance to stretch and between points with one vertex between them like A and C we require $2L$ distance which will cause resistance to stretching with additional resistance to bending.

The rest of the changes to the rope characteristics were made by parameter changes to the model. Cable length

 ISBN 978-80-86943-41-1

was chosen to be 10 meters and the number of segments was chosen to be 100. This was done in order to create a large amount of short segments, required for visually realistic underwater simulation. With such models, any disturbance at one end of the cable will propagate slowly and will be damped by the surrounding water body.

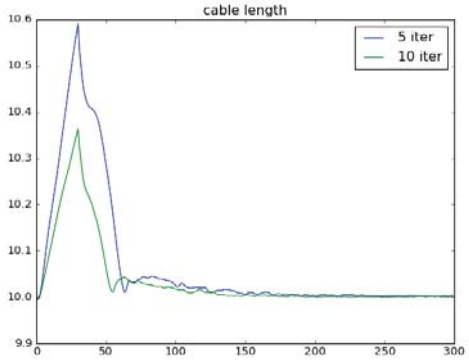

Figure 4: This figure shows the variation of the total cable length in meters with respect to the frame number. We can see that when using 10 solver iterations between frames to enforce the distance constraint of each cable segment the cable maintains its overall length more and represents a less stretchable cable.

The SolverIterations parameter (as can be seen in the pseudo code) should be chosen carefully. There is a trade-off between the stiffness or the ability to stretch and the damping mechanism introduced earlier. Since the damping is done in the Verlet stage, the constraint mechanism can still freely move all the rope particles, and due to that trade-off we limited the number of iterations. An improvement can be made to add some damping effect also in the constraint stage. That will allow more control on the cable length.

Setting the gravity to be zero was done to simulate the effect of neutral buoyancy. Usually, tethered systems are designed to meet the goal of neutral buoyancy to eliminate pulling forces from the cable in the case of non-neutral buoyancy. Additionally, the negative buoyancy of a tethered system can cause the cable to be tangled with objects on the surface of the seafloor.

Drag coefficient was chosen to be 0.9 and this value is much lower than the maximal value 1. This was done to introduce intense damping and to reduce the propagation along the cable. The random displacements coefficient was chosen empirically to be 0.1 and can be adjusted to different sea conditions. All the parameters of the simulation included the added parameters (the damping and the random displacements) can be controlled by the outside environment (like the game engine editor) and can be adapted to different types of cables with different characteristics.

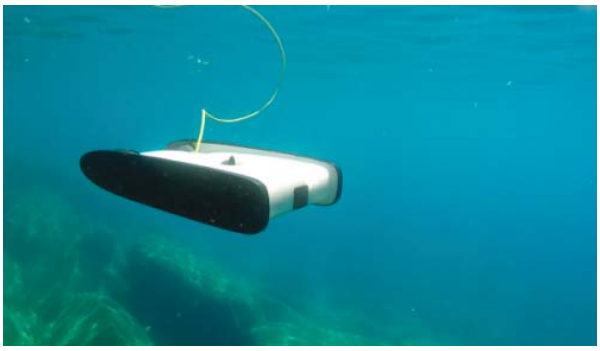

Figure 5: A small underwater OpenROV robot connected through a thin cable for video and control transmissions [16].

In our tethered ROV simulation, we have also made additional assumptions that there are no forces or relatively small forces introduced by the rope which effect the ROV position. In some cases, it makes sense for example if the mass of the ROV is relatively much higher then the mass of the rope. For example the OpenROV 5 [16] robot uses a very thin cable which handles only communication (not power) and in this case, we can assume that unless the cable is fully extended the relative force applied by the cable is relatively small. In practice, This means that the rope is not limiting the ROV movement.

3.1 Variable Length Cables

Underwater simulation of ROVs will also require modelling of cables connected to a spool that are released or retracted according to a naive logic that whenever there is a tension in the cable the cable is released. In the following, we outline a method to extend our model to a generate a variable length cable.

A flag is associated with each vertex of the cable model, and it represents whether the vertex is free to move according to the Verlet integration and the constraint mechanisms. By default, both ends of a cable will be flagged as non-free and the rest are free, since the cable is attached to both ends. In the case of a spooled cable, all the particles of the rope that are currently not released are flagged as non-free particles.

Since our model is position based, whenever the first segment from the spool side is stretched enough, typically by 10 percent of the total length, we will release a particle/vertex. After that, the Verlet integration and the constraint mechanism will move into action and will adjust the particles accordingly. In video [4] we can see that when the robot is moved the cable is pulled as necessary to maintain low tension.

We have made further modifications in the model, particularly in the areas closer to end points. Random forces were not be applied on the first free segment, to

ISBN 978-80-86943-41-1

avoid the spontaneous release of the cable due to random change of the length of the first free segment.

The spooled cable extension is done with the intention to lay a simulated foundation for the development and testing of managed tethered system. The simulation can report in real-time the current length of the cable and the estimated tension of the cable at each point along the cable. Specifically, in the beginning and the end of the cable wherein a real system we can place tension sensors as an input to the controller of the tethered system. The tension can be measured as a function of the distance between every two particles.

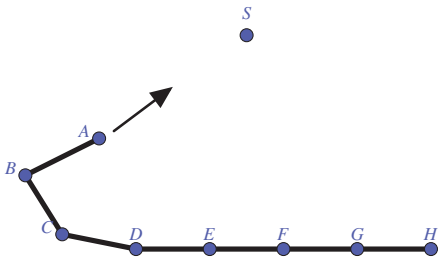

Figure 8: A 2D model of a flexible cable where one of the end points A is moved with a constant velocity v towards a target S

4 METHODS AND TOOLS

Figure 6: Unreal Engine 4 editor environment.

Figure 7: Experimental verification of motion of an extensible cable. We colored the cable in a checkers like pattern to enable the extension of the cable to be observable.

The main aim of this work has been to generate a convincing and realistic behaviour of an underwater tethered robot using the simulation framework provided by the Unreal Engine 4 (Figure 6). A live video demo can be seen in videos [17] [4] and in Figure 7. The experiments were done in the editor environment (not as a packed game). The robot seen in those figures was moved manually while the cable was attached to both ends. The new plugin is maintained and can be downloaded from here [18].

In addition, to have finer control over the simulation, an additional 2D simulation was created to demonstrate the proposed method. We used the Jupyter [19] python notebook environment to generate the output seen in figures 9 and 11. The 2D simulation is maintained under the following link [20]. Figure 8 illustrates the cable configuration used in the 2D simulation.

5 EXPERIMENTAL RESULTS

We created a simple 2D computer simulation to simulate the effects of moving one edge of the cable while the other end is pinned (Figure 7). Figure 9 shows the behaviour of the rope with and without damping with respect to time. The wave motion continues to propagate through the rope when there is no damping whereas with damping the wave energy slowly decays and random forces are becoming more dominant. In Figure 10 we can see our desired effect when using coefficient 0.9. The movement of the cable at one end does not affect the other end, so long as there is no tension in the cable segments.

Figure 11 shows the the random forces effect. In this experiment we look at the cable configuration in the 2D space after the system is stabilized ($t >> 0$) with and without random forces. We can see that the random forces create a kind of memory loss effect of the shape of the cable. This effect is important when there are no other significant forces (or they are close to zero) in the system. When we don't add the random force, the cable tends to stand still in contrast to what would be expected in a dynamic aquatic environment.

ISBN 978-80-86943-41-1

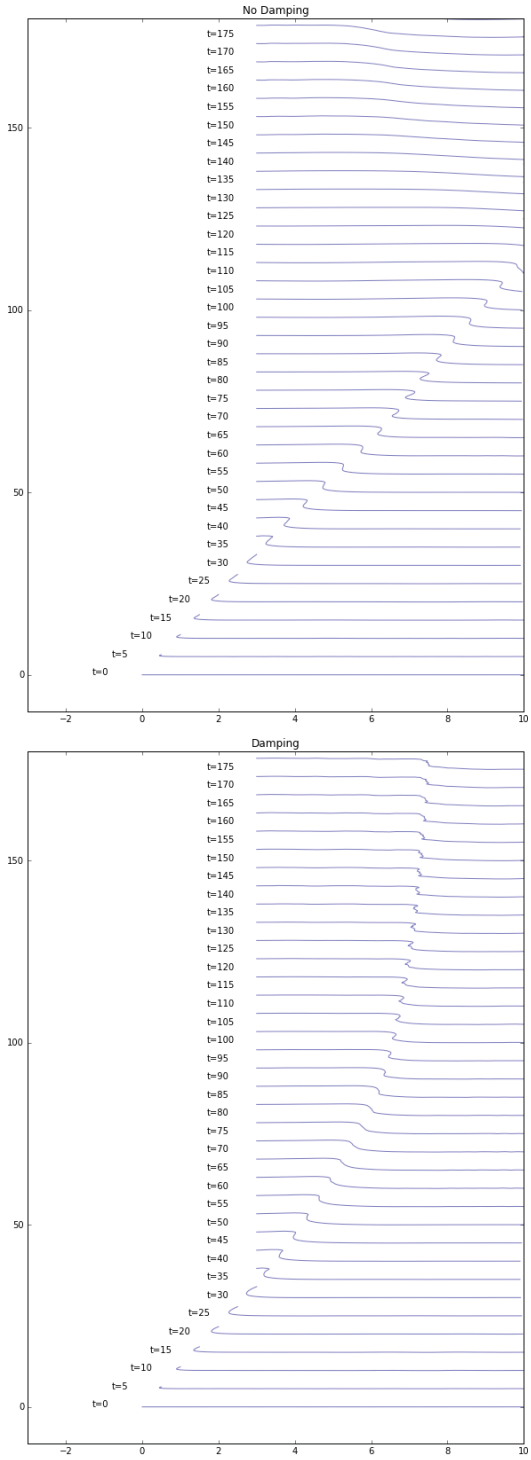

Figure 9: The damping effect. This figure shows the cable in different times. In t=0 we start to move the edge of the cable in the direction up and right along the "xy" plane. The first figure shows the results without damping and the second shows the behaviour of that cable with damping coefficient of 0.98. We can clearly see that the damping is absorbing the wave energy as we would expect in aquatic systems.

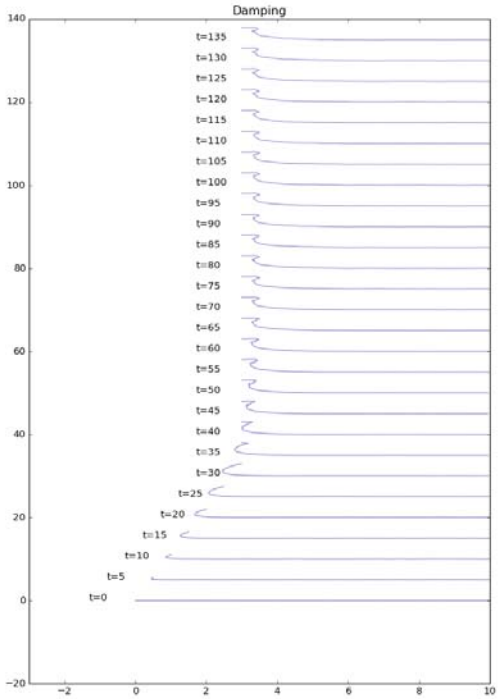

Figure 10: Damping with 0.9 coefficient. Tunning the coefficient to 0.9 causes the desired effect for underwater simulation in the case of a low tension cable in an underwater environment. Disturbance on one side remains local.

6 CONCLUSIONS AND FURTHER RESEARCH

In terms of performance, the modifications to the current model didn't require more computational effort. In fact, if we assume neutral buoyancy of the cable we can remove the gravitational forces from the simulation to reduce computational time. This can be useful in cases where a large number of cable/rope like object are simulated. Generally speaking, we can say that a cable is a linear 3d object curve which can be efficiently computed by modern CPUs.

Using the position based approach allowed us to easily modify the current model by adding drag and random displacements and in the future to apply other constraints for inter-rope tangling and ROV interactions. Further research will also deal with the forces applied to and by the cable to the objects that it is connected to. This can be done by measuring the length of the segments as described in section 3.1. Currently, we assume that there are no forces and torques applied by the rope which effect the ROV movement, and so we can improve future simulation by adding those forces to the simulation. Additionally, we added simple ran-

ISBN 978-80-86943-41-1

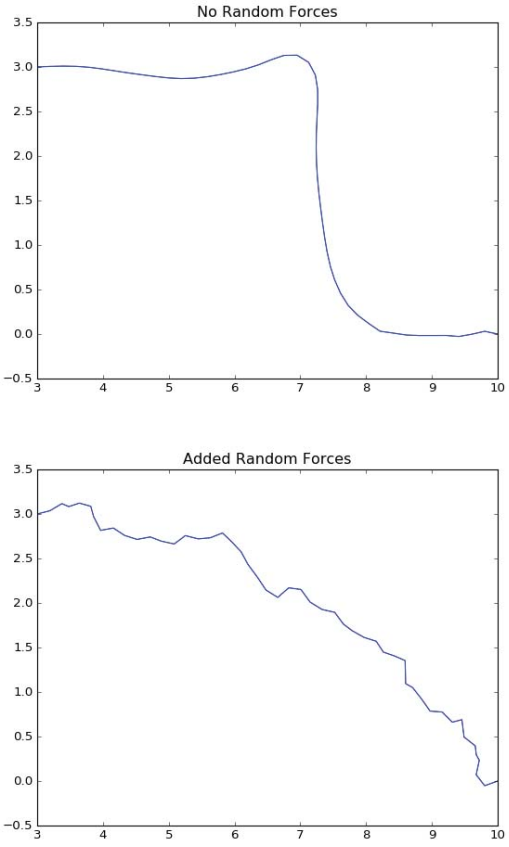

Figure 11: Introducing random forces to the system. Both images show the cable state after the system is stabilized (t»0). The first and the second images show the cable state with and without random forces respectively. We can see the "Memory Loss" that we would expect to see in a marine-like environment with underwater currents.

dom displacements with even normal distribution for all the segments. In real environments that is usually not the case and the currents are influencing the cable differently for each segment. Underwater currents are more similar to air turbulence and do not contain high frequency changes.

In our research, we found this simulation to be particularly useful in cases were the robots sees its own cable as presented in Figure 12. This fact may disrupt computer vision algorithms - especially those based on tracking using landmark features from the images and assume that these landmarks are not moving in the scene. With this new type of simulation, that kind of behaviour can be simulated in a manner close to real cable behaviour. New computer vision algorithms can be developed to mitigate that behaviour and new control algorithms can be developed to manage the cable configuration.

Figure 12: An underwater robot's camera view of its own tether cable

In this paper, we demonstrated a visually convincing underwater cable simulation. The current state of the art model implemented in the latest version of the Unreal Engine 4 was thoroughly investigated and the needed modifications to the model for underwater simulation were described in detail. We presented a novel approach to the underwater simulation and the unique characteristics of such a medium. We showed results in a 2d computer simulation for finer analysis of the simulation results. The simulation is robust and controlled by a large number of parameters as previously described. Finally, the modified model was implemented in Unreal engine 4 as a new underwater cable component available for download with provided demos demonstrating the new cable behaviour [18].

7 REFERENCES

[1] J. Bender, M. Müller, M. A. Otaduy, M. Teschner, and M. Macklin, "A survey on position-based simulation methods in computer graphics," in *Computer graphics forum*, vol. 33, no. 6. Wiley Online Library, 2014, pp. 228–251.

[2] M. Macklin and M. Müller, "Position based fluids," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 4, p. 104, 2013.

[3] T. Jakobsen, "Advanced character physics," in *Game Developers Conference*, vol. 3, 2001.

[4] Underwater cable reel simulation video. https://youtu.be/DO-x2RaZHso.

[5] R. Marshall, R. Jensen, and G. Wood, "A general newtonian simulation of an n-segment open chain model," *Journal of Biomechanics*, vol. 18, no. 5, pp. 359–367, 1985.

[6] B. Buckham, F. R. Driscoll, and M. Nahon, "Development of a finite element cable model for use in low-tension dynamics simulation," *Journal of*

ISBN 978-80-86943-41-1

Applied Mechanics, vol. 71, no. 4, pp. 476–485, 2004.

[7] Z. Wang and G. Sun, "Parameters influence on maneuvered towed cable system dynamics," *Applied Ocean Research*, vol. 49, pp. 27–41, 2015.

[8] C. Lambert, M. Nahon, B. Buckham, and M. Seto, "Dynamics and control of towed underwater vehicle system, part ii: model validation and turn maneuver optimization," *Ocean engineering*, vol. 30, no. 4, pp. 471–485, 2003.

[9] F. González, A. de la Prada, A. Luaces, and M. González, "Real-time simulation of cable payout and reel-in with towed fishing gears," *Ocean Engineering*, vol. 131, pp. 295–307, 2017.

[10] C. Ablow and S. Schechter, "Numerical simulation of undersea cable dynamics," *Ocean engineering*, vol. 10, no. 6, pp. 443–457, 1983.

[11] J. Burgess *et al.*, "Bending stiffness in a simulation of undersea cable deployment," *International Journal of Offshore and Polar Engineering*, vol. 3, no. 03, 1993.

[12] J. Gobat and M. Grosenbaugh, "Time-domain numerical simulation of ocean cable structures," *Ocean Engineering*, vol. 33, no. 10, pp. 1373–1400, 2006.

[13] S. Prabhakar and B. Buckham, "Dynamics modeling and control of a variable length remotely operated vehicle tether," in *OCEANS, 2005. Proceedings of MTS/IEEE*. IEEE, 2005, pp. 1255–1262.

[14] Cable component in unreal engine 4. https://docs.unrealengine.com/latest/INT/Engine/Components/Rendering/CableComponent/.

[15] L. Verlet, "Computer" experiments" on classical fluids. i. thermodynamical properties of lennard-jones molecules," *Physical review*, vol. 159, no. 1, p. 98, 1967.

[16] Openrov. https://www.openrov.com/.

[17] Cable simulation video. https://youtu.be/_QoMUSlQCsg.

[18] Cable sim project. https://github.com/UnderwaterROV/UWCableComponent.

[19] Jupyter. http://jupyter.org/.

[20] Cable sim notebook. https://github.com/UnderwaterROV/underwaterrov/blob/master/notebooks/rope.ipynb.

 ISBN 978-80-86943-41-1