Test Methods for Image-Based Information in Next-Generation Manufacturing

Henrique Oyama * **Dominic Messina** * **Renee O'Neill** *
Samantha Cherney * **Minhazur Rahman** *
Keshav Kasturi Rangan * **Govanni Gjonaj** * **Helen Durand** *

* *Department of Chemical Engineering and Materials Science, Wayne State University, Detroit, MI 48202 (corresponding author e-mail: helen.durand@wayne.edu).*

Abstract: Typical control designs in the process systems engineering literature have assumed that the primary sensing methodologies are traditional instruments such as thermocouples. Digitalization is changing the landscape for manufacturing, and data-based sensing modalities (e.g., image-based sensing) are becoming of greater interest for plant control. These considerations require novel test/evaluation solutions. For example, process systems engineering researchers may wish to test image-based sensors in simulation. In this work, we provide preliminary thoughts on how image-based technologies might be evaluated via simulation for process systems.

Keywords: Image-based control, cyberattack, camera sensor

1. INTRODUCTION

Over the last decades, significant advances in image-based sensors and image processing algorithms have occurred. This enables consideration of image-based control (IBC) systems in real-time process control based on camera sensors to perceive the environment and measure variables that would otherwise be impractical or time-expensive to measure. These advances have contributed to the increased exploration of IBC systems, including applications in robot manipulators Su and Zheng (2011) and bioprocess monitoring Höpfner et al. (2010). In the context of visual feedback control, different control architectures integrated with a vision system have been utilized to improve the performance of autonomous systems. In Su and Zheng (2011), for example, an image-based transpose Jacobian proportional-integral-derivative (PID) control was proposed for asymptotic regulation of robot manipulators with vision-based feedback. Image-based visual servoing control using a nonlinear model predictive controller (MPC) was proposed in Lee et al. (2011) as a vision-based obstacle avoidance strategy in a dynamic environment for an unmanned aerial vehicle. In the chemical industry, real-time image analysis systems have been installed and tested, for example, to monitor an industrial boiler system Yu and MacGregor (2004) or estimate bubble size at a phosphorus oxide flotation process Lin et al. (2008).

Although image-based control has been performed and tested with real systems involving camera sensors Lin et al. (2008), chemical processes are often large-scale and complex, making it challenging to visually replicate a next-generation manufacturing environment for the process industries without obtaining data from an actual plant. In light of this, it can be more difficult for process systems engineering researchers to test how new next-generation manufacturing concepts such as image-based control algorithms might fare in an actual plant. In, for example, the autonomous driving literature, it is common to discuss the simulation of virtual worlds and camera models (e.g., Rong et al. (2020)). It would be interesting to consider the use of virtual environments and simulation in a next-generation chemical manufacturing context, beyond education and training Ouyang et al. (2018). In this study, we provide a perspective on a number of applications for the chemical process industries where the computer graphics software toolset Blender may enable the generation of images as part of the determination of closed-loop control and safety system designs. Some of these applications include evaluating the impacts of cyberattacks on image-based controllers and considering humans at a plant in evaluation of control/safety strategies. This work thus serves as a preliminary discussion of a number of concepts for testing an image-based control framework for next-generation chemicals manufacturing.

2. EVALUATING PLANT IMAGE-BASED TASKS USING BLENDER

Blender is a computer graphics software toolset often used for computer animation. It is open-source and possesses many powerful capabilities in areas such as 3D modeling, rendering, and video editing. It therefore serves as an interesting framework for testing a variety of algorithms for next-generation manufacturing. This section presents several preliminary ideas in this direction, including its uses for image-based control evaluations when a camera sensor is provided false images by a cyberattack and for incorporating human factors in simulation. Though further research is needed to fully investigate the potential of Blender in such applications and compare it with other software, this section shows various ideas for how next-generation manufacturing technology could be represented and simulated by the process systems engineering community.

2.1 Evaluating Camera Sensor Cyberattacks on Image-based Control Systems Using Blender

Similar to other cyber-physical systems that involve communication channels and control architectures, IBC systems may also be subject to cyberattacks. Having a simulation-based framework to test the impact of such attacks would be desirable. In this direction, we investigate Blender's ability to provide insights into impacts of false camera sensors on IBC systems via a chemical process example of a level control. The level control process is represented below:

$$\frac{dh}{dt} = (u - c\sqrt{h})/A \tag{1}$$

where the state/controlled variable is the level of the tank h and the manipulated variable is the volumetric flow rate u entering the system. $F_{out} = c\sqrt{h}$ is the volumetric flow rate exiting the system, $A = 0.23$ m^2 denotes the cross-sectional area of the tank and $c = 0.008333$ m$^{5/2}$/s is the outlet resistance coefficient. The minimum tank level is 0 m and the maximum tank level is 0.5184 m. In addition, the minimum value of u is $u_{min} = 0$ m^3/s (which corresponds to a fully closed valve), and the maximum value of u is $u_{max} = 0.6$ m^3/s (which corresponds to a fully open valve). The process model represented by Eq. 1 is numerically integrated using the explicit Euler method with integration step of 10^{-3} s.

In our prior work Oyama et al. (2022), we developed an image-based control setup for the level control problem in the absence of a cyberattack. Specifically, we began by developing a representation of the tank level in the 3D viewport of Blender 2.93, so that images of the system could be generated using Blender's rendering capabilities and then processed using the Python Imaging Library (Pillow) Clark (2015). For this simulation, a plane was used to represent the tank level, with the tank assumed to be transparent and the fluid in the tank assumed to be a dark color. The initial level in the tank was $h(t_0) = 0.1$ m, and a proportional-integral (PI) controller was designed to drive the tank level to its set-point $h_{sp} = 0.4$ m over 7 s of operation. The PI controller has the following form:

$$\frac{d\epsilon}{dt} = h_{sp} - \tilde{h},\ \epsilon(0) = 0 \tag{2}$$

$$u = u_s + K_c(h_{sp} - \tilde{h}) + \frac{K_c}{\tau_I}\epsilon \tag{3}$$

where u is the controller output ($0 \leq u \leq 0.6$ m^3/s), $u_s = 0.0026$ m^3/s is the steady-state value of u that corresponds to the initial level of the tank at $t = 0$, ϵ is the dynamic state of the PI controller, and $\tilde{h}$ is the measured level of the tank. The value of $\tilde{h}$ was estimated from the rendered image of the tank level by using a color differential between the color of the fluid in the tank and the color of the background above the tank. Several RGB values were considered to define the top of the tank to account for differences in the RGB values at the top of the image due to lighting. When the color was no longer in the appropriate range, the index value I_p indicative of the position of the pixel in the image from the top (where larger values are closer to the bottom of the image) was recorded. It was then used in the following linear relationship, derived based on the pixel indexes I_p of the bottom and top of the tank image at $t = 0$, to convert the index at the top of the tank image to the level in the tank:

$$\tilde{h} = -3.8462 \times 10^{-3} \times I_p + 4.1577 \tag{4}$$

The bottom edge of the tank was positioned at (0,0,-1.57 m) in the Blender global coordinates. The tank level was modified by adjusting the height of the top edge of the tank (e.g., the tank level at $h = 0.1$ m is at a z-coordinate of -1.47 m). With tuning parameters of $K_c = 0.6$ and $\tau_I = 43.2$, the closed-loop state approaches the set-point after approximately 2 s under a sample-and-hold controller implementation with a sampling period of $\Delta = 0.1$ s. Even before attacks are added to the images, we can see some of the benefits of using Blender for assessing the image-based control algorithm before it is used in production. For example, for this case, it was noted that if it was desired to use RGB values as noted as part of the procedure for determining $\tilde{h}$, the lighting made it necessary to allow several RGB values to represent the top of the tank for use in determining I_p (instead of only one value for each of the red, green, and blue channels). In particular, the way for which the pixel index at the top of the tank level is acquired at every sampling time is the following: an index counter (I_c) is placed at the bottom of the tank level, which starts at index 1079. Then, the index counter increases by increments of one until a change in the RGB value from the color that represents the tank level to the color outside the boundaries of the tank level is detected. The RGB value outside the boundaries of the tank level ranges from rgb(29,65,19) to rgb(30,66,20), including any individual channel combination (i.e., red channel could be 29 or 30, green channel could be 65 or 66, and blue channel could be 19 or 20). The pixel index is then defined as $I_p = I_c + 1$. The fact that this range of RGB values was important for selecting the tank level shows that Blender can aid in understanding how an image processing algorithm interacts with the scene at hand for the control system design. Blender can also aid with tuning of the sampling period; for example, shorter sampling periods allow less visual change in the level before the next measurement is made, and Blender could aid in gaining a better sense of how the control law execution and image sensing interact.

In addition to aiding with understanding how control systems might behave in the absence of attacks, Blender can also aid in better understanding the impacts of attacks that might occur on image-based control systems, and how those could impact the process. To see this, consider the same closed-loop system described above but with disturbances and where a false image is provided as the sensor measurement/feedback to the image processing algorithm at every sampling time after 4 s of operation. Specifically, the tank image from $t = 0$ is provided to the image processing algorithm after 4 s of operation. In this case, the tank level measurement is lower than it actually is, causing the controller to compute control actions that attempt to increase the tank level. This causes the actual closed-loop state to continue to increase, which in a physical system could eventually result in overflow of the tank if the attack is not flagged (Fig. 1). With an image-based control system testbed in Blender, one could consider how changes to the image processing algorithm impact the success of attacks, or how detection strategies

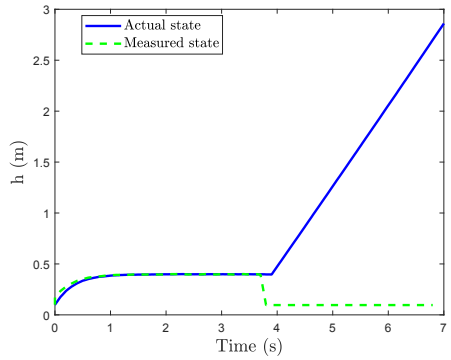


Fig. 1. Attack policy and closed-loop trajectory over time under the IBC system.

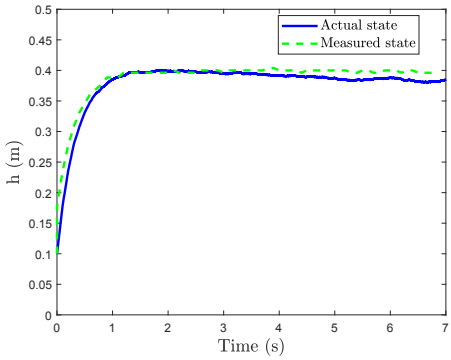


Fig. 2. Stealthy attack policy and closed-loop trajectory over time for the process under disturbance.

might be designed that can account for some of the complexity of the the situation in terms of variables such as lighting, cameras, occlusion, time, and color.

In addition, the effects of other cyberattack policies besides a constant fake image could also be examined. For example, Oyama et al. (2021) discussed a "stealthy" attack on traditional sensors in which false state measurements were developed by an attacker with full process knowledge who could generate a false state trajectory following the process dynamics and with disturbances taken from the same distribution as the actual plant/model mismatch, but different realizations. These attacks were considered to be "stealthy" in the sense that there would not be a way to distinguish them from the correct process state trajectory based on the trends in the process data only. One could investigate the development of stealthy attacks on image-based sensing as well. For example, we consider a stealthy attack in which false tank images are generated that would follow the process dynamics under the same disturbance distribution (but different realizations). To simulate this scenario, we consider now a case where the actual process dynamics of Eq. 1 are modified to include a disturbance added to the right-hand side of the differential equation describing the rate of change of h. This disturbance has zero mean and a standard deviation of 1 m/s, and bound of 0.1 m/s (this was also used in Fig. 1 but there with a standard deviation of 0.08 m/s). A stealthy attack is performed after 1 s of operation by simulating an auxiliary system with the same dynamics except that the realization of the disturbance is allowed to be different than in the actual process. Images corresponding to the level variations with these falsified dynamics are then generated and provided to the image processing algorithm for use in selecting the level. Fig. 2 shows the stealthy attack trajectory and the actual state over time in this case over 7 s of operation. We can observe that the controller drove the closed-loop state to the set-point and maintained it around that value until 5 s of operation. After that time, the closed-loop state starts to diverge from the set-point before the end of the 7 s simulated. These simulations show the power of Blender for enabling multiple scenarios to be analyzed with an effectively 2D image-based control system.

The ability to explicitly consider the image processing strategy enables assessment of the extent to which image-based control algorithms may fit within a control-theoretic cybersecurity framework. Specifically, prior work in our group has investigated techniques for integrated control and cyberattack detection policies (e.g., Oyama and Durand (2020)) under the assumption that process state measurements (with some noise) were available. In the example above, we see that despite the nontraditional sensor measurements being available, the image processing algorithm translates the image into an approximation of the process state $\tilde{h}$, which can make these state-based cybersecurity results applicable. Furthermore, Eq. 4 could be used to estimate the level for images of different heights of the tank and then to see what the maximum difference is between the measured level using this equation and the actual level under the assumption that no objects can come between the tank and the sensor, or that no other lighting or environment differences (e.g., fog) could occur that throw off the baseline image processing algorithm. To lower the level of measurement noise for control-theoretic results, one could move the camera closer to the object so that the difference in level between two pixels is smaller. If issues such as occlusion or unexpected environmental conditions could occur, however, this opens up new types of stealthy cyberattacks in which attackers could add plausible obstructions to the image to attempt to throw off the measurements. This would be a key difference between traditional measurements and image-based measurements from a cybersecurity perspective (because it would then suggest that a rapid decay of the fidelity of a measurement would not be indicative of abnormal behavior at the sensor). These insights can be used to aid in the design and placement of camera sensors within a plant.

Remark 1. No attempt was made to create a photoreal image in the simulation of the level controller (the fluid in the tank was taken to be black with a dark green background used to represent the system). One might ask how chemical engineers might move toward more photoreal simulation of a plant. This has been considered in, for example, virtual reality simulation for chemical plants Schofield (2012). For the goal of using photoreal simulations for evaluating control and safety system designs, however, one of the challenges would be that virtual environments might be developed regularly for research on different systems, so that a methodology for moving from literature on a specific process to a 3D model of this process (likely in the absence of industrial data on a specific plant layout) would be needed. This would need to proceed through several stages, including a stage in which a plant description is translated

into a basic plant layout in space (to identify where units should be situated with respect to one another), another in which the units in this layout are modeled in greater detail in 3D space (to, for example, capture typical geometries more along the lines of computer-aided design), and another in which the lighting and materials (from an image perspective where a material interacts with light to create a specific color/texture that can be viewed) for these units are finalized. Each of these steps poses new challenges for a typical process engineer. For example, even the first step that is most directly tied to plant design requires thinking about a plant with significant depth. To see this, consider a polyethylene terephthalate (PET) production process. PET production occurs in three distinct steps: esterification, polymerization, and solid-state polymerization (SSP). At least two esterification reactors are required to produce the monomer, bis-hydroxyethyl terephthalate (BHET) with significant conversion of the terephthalic acid (TA) and ethylene glycol (EG) reactants; we consider a series of batch reactors (the primary esterification reactor (PE) and the secondary esterification reactor (SE)). Polymerization puts together these monomers into short chain polymers, whereas in SSP the short chain polymers are joined into their final long chain form. A typical process systems engineering approach to designing such a process is to consider translating the major steps in the literature for such a process into a series of unit operations that can be incorporated in process simulation or computational fluid dynamics simulation software. Moving a full process from the literature to a 3D plant environment requires consideration not only of the basic unit connectivity, and even features of individual unit orientation (e.g., one of the reactors in the PET process is tilted because it allows for the contents to move towards the outlet of the reactor by both the force of gravity, and the rotational forces exerted by the shaft Im et al. (2017)), but also of auxiliary piping and its placement throughout the plant (e.g., nitrogen passes over the PET pellets in the reactor countercurrently to remove the byproducts from the SSP reaction and helps regulate the SSP reactor temperature Kim et al. (2003), which means nitrogen piping would need to be considered in a 3D plant). While Blender has the ability to enable 3D geometries to be developed with features like tilts to some units or continuous segments that could represent piping (e.g., Fig. 3), the use of this software by chemical engineering researchers to represent plant environments would require chemical engineering researchers to postulate details of plant construction while also learning to manipulate lighting and visuals so that image-based control algorithms could be examined in a context more likely to be representative of a physical situation.

Remark 2. As noted above, the level control example utilized a 2D approximation of level in a tank for a clear tank for the control policy. However, Blender in general is capable of 3D simulation, and therefore can be used for more complex image-based control simulations through a combination of physics modeling (which can be simulated in Blender's Python programming interface) and spatial modeling. To showcase this idea, consider a single sphere moving in space (inspired by image-based control of zinc flotation Kaartinen et al. (2006), this might be considered to represent a bubble in a froth) with a "sun" light source in Blender, and the camera set to track the sphere/bubble

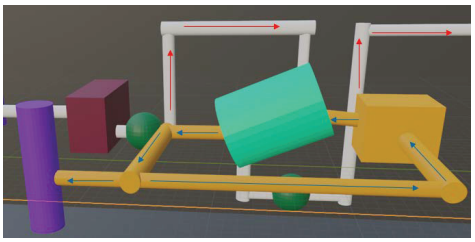


Fig. 3. Blender 3D environment demonstrating tilted cylindrical geometry and cylindrical geometries with a look of piping.

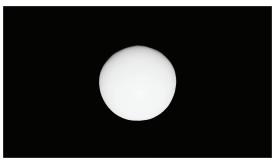


Fig. 4. Bubble render.

as it moves. To show movement of the bubble over time, "keyframes," which are times at which a specific feature of an object's location, rotation, or scaling are enforced so that movement between the prior keyframe and the new keyframe is interpolated by Blender to create an animation, are set at frames 1, 10, 20, 30, 40, 50, 60, and 70, and the sphere locations are set in these instances to (0,0,1), (2,0.5,1.5), (2.5,0.5,2), (3,0.5,2.5), (2.5,0.5,3), (2,0.5,4), (1.5,0,5), and (1,-0.5,6), to trace out a path for the sphere over time. By frame 90, the sphere is assumed to be at a final desired position (this might correspond to the top of the froth in a more complex simulation), and then the image is taken by the camera via a rendering (Fig. 4) and printed to a file. To show that the image can be processed to cause a "control action" to then be taken on the moving sphere based on the image, the width and height of the image are obtained, and if the number of pixels in the width times the number of pixels in the height is less than 10000000, the sphere is dropped. Though this discussion does not yet showcase a chemical process-relevant control simulation, it indicates that an object in 3D space can be manipulated, images generated after an object moves, and then an action taken on the system as a result. Completing the simulation of the frothing process includes synthesizing a visualization of bubbles, including how the zinc particles cause the bubbles to "look" in the presence of the light source, with differential equations describing physics such as attachment of particles to bubbles Do (2010).

2.2 Accounting for Human-Plant Interaction Using Blender

Besides simulation of chemical systems where visuals are important, Blender can be used to evaluate how humans interact with plants, control, and safety. For example, one might wish to generate data corresponding to people moving about a plant and carrying out tasks to analyze how different algorithms for evaluating safety of individuals at a plant using camera sensors may work. If limited data of people interacting with the system is available, or if the situation it is desired to evaluate does not exist in real life yet, Blender could be used for generating many potential human movements. A rough example of a human model

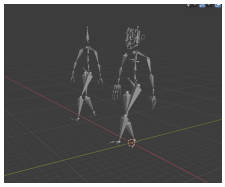
is shown in Fig. 5a, with the meshing shown on the left. This human can then be animated to perform a number of different actions at the plant (for example, Figs. 5b-5c shows the human walking and turning a valve). Other capabilities, such as an add-on in Blender for retargeting a mesh, can be used to map approximately the same animation to a new body (Figs. 6a-6b). This might be used for generating many of the same tasks being performed by different human models. To create Figs. 6a-6b, the root bone (spine) and other bones were retargeted from the original human onto a new figure. Fig. 6a shows a case in which there are similarities in the motions of the figure to which the data was retargeted compared to the original, but where the retargeting is not perfect and there is misalignment of bones. This leads to the animations not being identical, which one might argue could have value for assessing how perturbations of the original movements affect any assessments being tested with the animations. For example, the retargeted model in this case does not have as wide a range of motion as the original, so that if it was desired to test whether the original human would have passed a plane where the alarm sounds, the original human would have caused that alarm to sound, whereas the retargeted human would not (for example, in Fig. 6b). One could also imagine using animation in Blender to generate many different movements that humans might take in a certain area of a plant and assessing how different algorithms for flagging potential safety breaches react to the different movements.

(a) Human model.

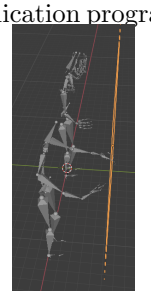
(b) Human walking.

(c) Human turning valve on and off.

Remark 3. In the discussion above, many positive characteristics of Blender were highlighted, but it should be compared with other software that also enables rendering of images in future work before deciding on a framework or software workflow for image generation and human interaction modeling for simulation in a chemical process context. It should also be highlighted that there are applications that one could imagine in a next-generation manufacturing context where detailed graphics manipulation (e.g., through use of the application programming interface



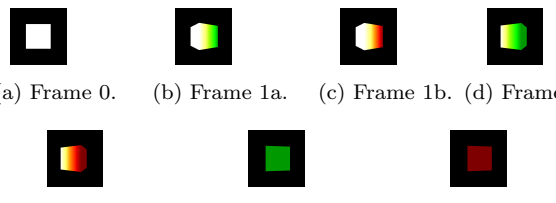
(a) Original and retargeted meshes.



(b) Human recording data.

OpenGL) may be preferred over the use of Blender. As a thought experiment in this direction, consider the case where one might want to use prediction of the future visual "state" of a system in a predictive control law. One might imagine investigating whether this has benefits using a context similar to our group's work in Oyama and Durand (2022), where it was considered that there were multiple possible models of a system's behavior (in this case described by differential equations) and that it was desired to use a controller to aid in ascertaining which was most consistent with the dynamics as quickly as possible. One might ask whether there could be cases where multiple possible postulates about the "look" of part of an environment could be considered when the camera view is limited, and where an autonomous agent (e.g., a robot taking measurements at a process plant) would want to take a path that is most rapidly instructive to it in revealing which of its possible postulates about its environment's "look" is correct. If it was desired to consider incorporating prediction of the future "look" of an environment as an autonomous agent moved in different ways in a controller for modifying the agent's movement, it may be of interest to attempt this with graphics programming to integrate it with the rest of the optimization algorithm code.

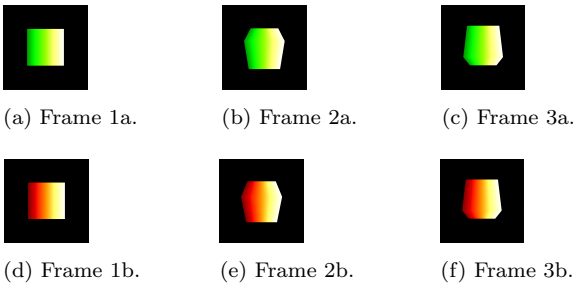
To illustrate the concept, consider a single cube with a black background where either the cube has a white to red gradient along the z-axis, or a white to green gradient. To convey movement around the cube, the starting image is transformed using model transformations (developed with guidance from Angel and Shreiner (2011)) calculated and applied using the computer graphics specification OpenGL and coded in C++, generating the rotations around the object's y-axis and z-axis, respectively, seen in Figures 7 and 8. The figures are generated by rendering and taking a screenshot of a still frame which represents a transformation forward in time. In each trajectory, both the position and angle of the camera are changed to keep focus on the center of the cube. In Figure 7, initialized with a head on view of a cube face colored white to make the first frame of each model indistinguishable, the camera's orbit around the y-axis begins to reveal more drastic differences in the two models as more of the color gradient is seen. In Figure 8), however, the orbit around the z-axis shows that the difference between the two models does not change substantially along that trajectory.



(a) Frame 0. (b) Frame 1a. (c) Frame 1b. (d) Frame 2a. (e) Frame 2b. (f) Frame 3a. (g) Frame 3b.

Fig. 7. Sequence 1, y.

The above results show that there are ways of moving around the cube that can result in a camera position that does not reveal much more about the "look" of the object than the prior position, and also those which may provide more details on the object's "look" (and therefore could be directions which one would hope a predictive



(a) Frame 1a. (b) Frame 2a. (c) Frame 3a.

(d) Frame 1b. (e) Frame 2b. (f) Frame 3b.

Fig. 8. Sequence 2, z.

controller might choose when seeking to better understand an environment).

3. CONCLUSION

This work discussed several applications of Blender for evaluating safety and control systems based on images or human interaction, and also highlighted that other methods for handling graphics (e.g., graphics programming) may be of interest as well for next-generation manufacturing applications.

4. ACKNOWLEDGEMENTS

Financial support from the National Science Foundation CBET-1839675 and CNS-1932026, the Air Force Office of Scientific Research under award number FA9550-19-1-0059, the National Aeronautics and Space Administration (NASA) under award number 80NSSC20M0124, Michigan Space Grant Consortium (MSGC), and Wayne State University is gratefully acknowledged. We would like to thank: 1) sociamix for the tutorial "Modeling a character BaseMesh in Blender (Tutorial)," which was followed for creating the human model; 2) the creators of the tutorial "Character Rigging- Blender 2.80 Fundamentals" for providing the YouTube tutorial used for rigging; 3) Sebastian Lague for the YouTube tutorial "RPG graphics E04: Walk animation" that enabled creation of the human walk cycle; 4) Rokoko for their Youtube tutorial "Rokoko Guide: Easiest Motion Capture Animation Workflow for Blender!" that enabled the motion tracking to be performed using the Blender data; 5) cgvirus for the Youtube tutorial "Retargeting Any Motion Capture Data in Blender From Mixamo BVH iClone to Rigify Daz3d or Custom Rig" that was used to map the captured data to a new rig; 6) thecherno for his OpenGL Youtube series for tutorials on coding essential OpenGL functions.

REFERENCES

Angel, E. and Shreiner, D. (2011). *Interactive Computer Graphics: A Top-Down Approach with Shader-Based OpenGL*. Addison-Wesley Publishing Company, USA, 6th edition.

Clark, A. (2015). Pillow (PIL fork) documentation. URL `https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf`.

Do, H. (2010). *Development of a turbulent flotation model from first principles*. Ph.D. thesis, Virginia Tech.

Höpfner, T., Bluma, A., Rudolph, G., Lindner, P., and Scheper, T. (2010). A review of non-invasive optical-based image analysis systems for continuous bioprocess monitoring. *Bioprocess and Biosystems Engineering*, 33, 247–256.

Im, S.K., Jun, S.M., Bae, S.H., Kwon, S.Y., Lee, E.J., Jin, Y.S., and Park, K.K. (2017). Continuous solid-state polymerisation device and method. US Patent 9,649,616.

Kaartinen, J., Hätönen, J., Hyötyniemi, H., and Miettunen, J. (2006). Machine-vision-based control of zinc flotation—a case study. *Control Engineering Practice*, 14, 1455–1466.

Kim, T., Lofgren, E., and Jabarin, S. (2003). Solid-state polymerization of poly (ethylene terephthalate). I. Experimental study of the reaction kinetics and properties. *Journal of Applied Polymer Science*, 89, 197–212.

Lee, D., Lim, H., and Kim, H.J. (2011). Obstacle avoidance using image-based visual servoing integrated with nonlinear model predictive control. In *IEEE Conference on Decision and Control and European Control Conference*, 5689–5694. Orlando, Florida.

Lin, B., Recke, B., Knudsen, J.K., and Jørgensen, S.B. (2008). Bubble size estimation for flotation processes. *Minerals Engineering*, 21, 539–548.

Ouyang, S.G., Wang, G., Yao, J.Y., Zhu, G.H.W., Liu, Z.Y., and Feng, C. (2018). A Unity3D-based interactive three-dimensional virtual practice platform for chemical engineering. *Computer Applications in Engineering Education*, 26, 91–100.

Oyama, H. and Durand, H. (2022). Lyapunov-based economic model predictive control for online model discrimination. *Computers and Chemical Engineering*, 161, 107769.

Oyama, H., Leonard, A.F., Rahman, M., Gjonaj, G., Williamson, M., and Durand, H. (2022). On-line process physics tests via Lyapunov-based economic model predictive control and simulation-based testing of image-based process control. in press. Atlanta, Georgia.

Oyama, H. and Durand, H. (2020). Integrated cyber-attack detection and resilient control strategies using Lyapunov-based economic model predictive control. *AIChE Journal*, 66, e17084.

Oyama, H., Rangan, K.K., and Durand, H. (2021). Handling of stealthy sensor and actuator cyberattacks on evolving nonlinear process systems. *Journal of Advanced Manufacturing and Processing*, 3, e10099.

Rong, G., Shin, B.H., Tabatabaee, H., Lu, Q., Lemke, S., Možeiko, M., Boise, E., Uhm, G., Gerow, M., Mehta, S., Agafonov, E., Kim, T.H., Sterner, E., Ushiroda, K., Reyes, M., Zelenkovsky, D., and Kim, S. (2020). LGVSL simulator: A high fidelity simulator for autonomous driving. In *IEEE International Conference on Intelligent Transportation Systems*, 1–6. Rhodes, Greece.

Schofield, D. (2012). Mass effect: A chemical engineering education application of virtual reality simulator technology. *Journal of Online Learning and Teaching*, 8, 63.

Su, Y. and Zheng, C. (2011). A simple PID control for asymptotic visual regulation of robot manipulators. *International Journal of Robust and Nonlinear Control*, 21, 1525–1540.

Yu, H. and MacGregor, J.F. (2004). Monitoring flames in an industrial boiler using multivariate image analysis. *AIChE Journal*, 50, 1474–1483.