Time Delay Based Dynamic System of Networked Autonomous Vehicles

Nicholas Gamez\(^1\), Student Member, IEEE, Prasanna Kolar\(^2\), Student Member, IEEE, Ahmad Taha\(^3\), Member, IEEE, and Mo Jamshidi\(^4\), Fellow, IEEE

Abstract—Present day large scale autonomous systems comprise of closed loop networked subsystems that need new scalable communication, control and computation techniques to interact with humans. In order to be stable and work efficiently, these systems need to be closed and networked. We have developed a control model and simulated this model to incorporate delays that are encountered when the subsystems communicate amongst each other. The simulations show the effects of time delays and the resulting instability from too much delay. The time delays are modeled for both communication and computation in a system of multiple vehicles.

I. INTRODUCTION

Large scale autonomous systems that contain closed loop communication networks are more common these days due to the cheap availability of many of the control, communication, sensory and actuator systems. These large scale autonomous systems (LSAS) can contain a variety of systems that utilize a time-delayed feedback control system. Its applications can be found in chemistry, biology, physics, engineering and also in medicine. Typically these systems do not require a reference system and needs a minimum a priori system knowledge. Gallardo, et. al. [1] have presented a formation control system with hardware of three UGVs and a UAV based on the leader-follower premise and a “virtual” leader on a Parrot Bebop drone and three Kobuki Turtlebot vehicles.

Time delay systems have also been used to control unstable periodic orbits, provide a tool to stabilize unstable steady states [2]. Hovel also applied time-delayed feedback to different classes of dynamic systems [3]. This enables dynamic and ultrafast systems as in optics and electronics, to utilize this control method[4]. In such systems the data is communicated, i.e., sent and received across a wide range of network, sensory, actuator and control systems. Sensors measure, process and communicate values across the network to the control/computation nodes, which in turn process them and communicate the values over to the actuator systems. The actuator nodes receive these new values and apply them to the process input. A control algorithm calculates the signals that are needed to be sent to the actuator nodes. An ideal system will perform all these instantaneously, but such systems are few and far in between since the LSAS systems are normally distributed in nature and delays are associated with such systems and these are the inter-system delays or latency. Latency is a time delay between the cause and the effect of some physical change in the system being observed or more technically is a time interval between the stimulation and response. In fact delays are encountered even if the system is not distributed, due to delays in communication within the same system’s sensor, control and actuator nodes. These are the intra-system delays. Models to mimic such intra-system and inter-system delays have been developed as part of this research. Our research demonstrates that time-delayed feedback control system is a tool that is powerful and can be efficiently used in the areas of mobile robotics involving an unmanned ground and aerial vehicles (UGV and UAV’s respectively).

There are several factors that affect the delay in a system. These are mainly dependent on the type of the module namely sensors, controllers, computers, actuators. Sensor delays are dependent on how fast a sensor can read (measure) the input data and/or process it and send it to the computer/controller. The delay in the computer is dependent on how efficient and capable it is, i.e., how efficiently can it process the information it is given. It also depends on the efficiency of the algorithm that is implemented and then communicated to the controller. The delay in the controller depends on how quickly it can apply the required control algorithms and send the processed control signals to the actuators. The delays in the actuators can be attributed to the efficient processing of the control signals. There are broadly two types of delays namely communication and computation. A system can have a combination of these 2 delays. The delays have a time dependency defined in steps of \( k \).

- Computational delay in the controller \( \tau_{k}^{c} \)
- Communication delay between the sensor and the controller \( \tau_{k}^{sc} \)
- Communication delay between the controller and the actuator \( \tau_{k}^{ca} \)

The complete system will have a total control delay for each time step \( k \)

\[
\tau_{k} = \tau_{k}^{c} + \tau_{k}^{sc} + \tau_{k}^{ca}
\]

The delays typically occur in a random manner, but we can list them as follows

- Random delay without correlation or stochastic delay
- Random delay with probabilistic correlation
- Constant delay
II. DELAY MODELING

The time delayed feedback system in its original form as introduced by Pyragas [5] is as follows:

\[
\frac{dx}{dt} = f(x(t), u(t))
\]

where \( x \) is a state vector of the \( n \)th dimensional state space while \( u \) is the control inputs of \( m \)th dimensionality of the system.

\[
x \in \mathbb{R}^n \rightarrow \mathbb{R}^n \text{ with } f : x \mapsto f(x)
\]

This measures the state \( x \) to create a control signal in the \( m \)-dimensional signal space. We consider \( \tau \) as the time delay. This control signal could be for instance, a single component of the state vector \( x \). The main part of Pyragas [5] control is to generate a control force, \( F \), that consists of the difference between the current signal \( u(t) \) and a time delayed counterpart \( u(t - \tau) \). In summary we find that there are several advantages to utilize a time-delayed feedback system. Some advantages are that there is no need to utilize a reference signal. The other advantages are the minimum knowledge of the investigated system and easy experimental implementation. The reference signal is not needed since the time-delayed feedback system itself generates the reference signal from the delayed time series of the system under control.

III. ENVIRONMENT

The test environment consists of vehicle models and system simulations. We will present both these environments in detail below in subsections Simulation and Implementation. We would like to point out that the simulation environment is a software environment using Matlab, Simulink, and Truetime [6].

A. Simulation

The simulation will be completed in a Linux and Windows environment using programming software like Matlab, True-time, simulink and plotting. All these together are used to render the processing and execution of the code graphically. The results section contains more details. This environment is executed on a laptop comprising of at least 4 GB RAM, 200 GB Harddisk and Quadcore processing power.

**Truetime** which is an additional package for Matlab/Simulink and has been used to simulate hardware components like a router, wireless network and computational delays. This versatile tool has given us an opportunity to simulate in the truest sense the hardware components. This usage will remove any doubts about the accuracy of the simulation compared to the hardware. TrueTime is a Matlab/Simulink-based simulator for real-time control systems. TrueTime facilitates co-simulation of controller task execution in real-time kernels, network transmissions, and continuous plant dynamics [6].

B. Implementation

An autonomous system of three vehicles is shown in Figure 3. In the depiction, a UAV drone is searching for any point of interest. Once it finds the point, here a vehicle on fire, the drone relays the coordinate position to the UGVs so they can provide assistance to the burning vehicle. This system is the basis behind the simulation modeling and results that follow.

Two goals were analyzed through simulation in Simulink

1) Analysis of time-delays in a single vehicle system
2) A system of systems with two vehicles coordinating together

For the initial simulation, a differential drive UGV was modeled in Simulink [7]. A differential drive, much like a car, cannot move laterally due to the nonholonomic constraint [8] derived from the dynamic equations shown in 2. The control/input variables are chosen to be the linear and angular velocities, \( u \) and \( \omega \) respectively.

\[
\begin{align*}
\dot{x} &= v \cdot \cos(u) \\
\dot{y} &= v \cdot \sin(u) \\
\dot{\theta} &= \omega
\end{align*}
\] (2)

Due to these constraints of no lateral movement and for simplicity, a three part controller was created to control the UGV from the distance and angle to a desired coordinate point. Given the initial position of the rover \((x_0, y_0)\) and a desired point \((x_{\text{ref}}, y_{\text{ref}})\), the UGV can be transformed from the states \(x, y, \theta\) to the system shown in equation 3, where \(\beta\) is the required angle to face the desired position, \(d\) is the absolute distance to the final position, and \(\alpha\) is the final orientation to face once the position has been reached.

\[
\begin{align*}
\beta &= \tan^{-1} \frac{y}{x} \\
d &= \sqrt{x^2 + y^2} \\
\alpha &= \beta - \theta
\end{align*}
\] (3)

For the two angle states (\(\beta\) and \(\alpha\)), \(\omega\) is the only input that must be controlled, while \(v\) is the only input required for distance, \(d\). After the transformation to the new states, each variable can be controlled in a cascading fashion with three separate state feedback controllers. The control law \(u = -K \cdot x\) is sufficient to successfully control the UGV to reach a desired point. This same UGV is used in the second simulation as well.

The model of the UGV is shown in Figure 4 with two transport delays. The first delay would correspond to the delay in transporting the transformed states from a microcontroller to the microcomputer where the control takes place. The second delay situated from the controller to the plant dynamics simulates the computational delay while applying the controller to determine the required system inputs. Multiple delays were analyzed by the time required to reach the desired reference point to find a time where the system becomes unstable. The results are shown in the following section.

With the UGV already modeled, a subsequent model for a quadrotor UAV was developed in Simulink. The UAV is a highly nonlinear system with three linear and three angular degrees of freedom. For this topic, the control of the UAV was limited to hovering above the working space of the UGV sending coordinates to the ground rover through a wireless network modeled with TrueTime [6].

The 12 states of the UAV are shown below:

- \(\phi, \theta, \psi\): The pitch, roll, and yaw angles respectively
- \(x, y, z\): The coordinates with respect to the earth’s reference frame
- \(u, v, w\): The linear velocities in the UAV’s inertial frame
- \(p, q, r\): The angular velocities in the UAV’s frame

Also, there are 4 inputs to the quadcopter that control the value of each of the 12 states:

- \(u_1\): Vertical Thrust
- \(u_2\): Angular motion along the X direction
- \(u_3\): Angular motion along the Y direction
- \(u_4\): Angular motion along the Z direction

To model a hovering UAV, the dynamics were linearized around an operating point where all states are zero with the exception of \(x, y, z\) coordinates. With the the linearization of the UAV, the system can be modeled in state-space format.

\[
\begin{align*}
\dot{x} &= Ax + Bu \\
y &= Cx
\end{align*}
\] (4)

The linearized model can be controlled with a state feedback matrix as well. A linear quadratic regulator (LQR) controller was used to determine the feedback matrix to successfully control the hovering of the UAV [9]. The controller created will bring the linearized UAV to a hovering point from resting position and hold the position while reading the image data for any points of interest to send to the UGV.

Figures 5 and 6 show the UAV simulation moving to the correct position in 3D coordinate space as well as the time it takes to move for each coordinate to be reached. The reference coordinate the UAV moves to is \((3.1, 8)\) taking close to 16 seconds to reach the desired height.

With both systems modeled, the goal is to connect them together as a system of systems with time delays between their communication systems. The UAV and UGV communicate

![Simulink model of a single UGV with Delays without UAV. Reference trajectory is generated randomly](image)
Fig. 5. UAV coordinates with respect to time

Fig. 6. UAV moving to coordinates (3, 1, 8) with LQR controller

with each other through a wireless network while the UGV communicates between its controller and actuators through a wired Ethernet network. This system is shown in Figure 7 with one UAV and one UGV. With Truetime, the wireless system can be modeled with a physical distance between nodes.

Within the system shown in Figure 7, there are three separate communication delays modeled.

1) The wireless network between the UAV to UGV and UGV to UAV
2) Ethernet network within the UGV emulating a microcontroller to microcomputer system on the UGV
3) Computational delay in the state feedback controller of the UGV

Different combinations of these three network delays were considered in the results section. The wireless network has a 50 Kbps data rate, and the Ethernet network was given a 1 Gbps data rate for the simulations. Finally the computational delay for the UGV controller was 0.5 seconds.

IV. RESULTS

UGV simulation results

With one system, there was no wireless communication and Truetime was not used to model the delays. The delays in Figure 4 are modeled with transport delay blocks where it functions as a wait below the data is passed to the next simulation step. For the system, the two delays are considered to be constant for this scenario and the results are shown in Figures 8 - 9 for the X, Y coordinates, and the orientation respectively.

The first interesting thing to notice is the 75 ms delays reach the desired trajectory faster than the system with no delays. When a delay is introduced, the control signals sent are carried out for a longer time. Consequently, the system may reach the desired point quicker, but there will be either a higher overshoot as seen in the orientation. This is very apparent when you have more than 100 ms delays. For a single point, the system can tolerate delays smaller than 100 ms and still reach the desired trajectory within 10 seconds. The higher the delay the more the system will create overshoot and longer destination times.

If obstacles were introduced into the system, the delays could create collision problems because the controller cannot send the correct input in time before the previous caused a problem. The more delay creates a choppier path towards the goal. Obviously the system cannot account for sharp turns if a new obstacle presented itself during the operation. The UGV begins to break down when two delays at 150 ms are introduced especially in the angle orientation. The system begins to oscillate more violently and never reaches the desired angle or position in 10 seconds.

UGV & UGV multi system simulation

Figures 10 - 13 depict the movement of the UGV which is given its destination coordinates by the UAV. Subsequently, the UGV sends a signal back to the UAV when it requires another destination. This system communication is modeled through a wireless communication module between the two
vehicles. As stated before, there are three different time delays associated with the system, and five different models were tested where different delays were considered:

- Wireless network delay between vehicles only
- Ethernet based delay within the UGV only
- Computational Delay for UGV controller only
- All 3 delays together
- No delays

The simulation is set up in such a way that data is only sent when needed from the UAV to UGV. Therefore, the wireless network is not saturated with information continuously and the delay associated is so small the system models very close to the original simulation without time delays. In Figure 10 and Figure 11 the dotted lines show the effect of just the wireless communication delay, which is very similar to the simulation with no delays. When the wireless network is not being continuously used, the delay propagation is small.

When communication delay for the UGV from its controller to the plant is considered, also shown in Figures 10 and 11, the system reaches the trajectory faster, but the movement is much choppier. Interestingly, when the communication delay is small much like in the first results, the new input magnitudes lag and the higher magnitudes move the UGV closer than anticipated. This results in it reaching the desired trajectory faster. An issue will occur if the delays are too big and the system begins to overshoot the goal.

For the first two delays separately, there is not a big change compared to the delay-less system. These systems, however can be pushed to instability with higher delays in either system.

Figures 12 and 13 depict two systems, one with the computational delay within the UGV’s controller and another with all three of the previous delays together.
The computational delay was stated to be 0.5 seconds long for the UGV controller, which again shows the same effect as the communication delay inside the UGV. The system moves faster to the position, but the angular movement oscillates and doesn’t reach its desired point. The computational delay creates an instability in the controller at 0.5 seconds.

The final simulation results stem from the system with all three of the previous delays included. The simulation here should mimic a real-time system the closest. For this system, the trajectory of the UGV takes longer to begin moving and travels less smoothly than the previous simulations. While the UGV still reaches the desired trajectory, it does not reach the required angular orientation. The oscillations increase much more than the system with just a computational delay.

For the system overall, we can conclude the delays associated with a 1 Gbps wired connection will decrease the settling time of a system when used continuously. A 50 Kbps wireless network affects the system minimally when only used to transmit data sparingly when needed, but it can be postulated to degrade the system considerably if used continuously. With all three delays, the system will still reach the desired endpoint with more time and less smooth movement.

V. CONCLUSION

Given a multi-vehicle system with time delays, the simulation results show the system reaching a state of partial equilibrium. The ground vehicles successfully reach the planned destination with multiple delays, but it does not reach the correct angular position when computational delays in the controller are considered. For the fully delayed system does not reach the required angular orientation as well, oscillating around the final point. The simulations show the effect of time delays in a multi-agent system with wireless and wired network components. For the given networks with sufficient bandwidth, the system can reach its desired trajectory.

Next steps are to find the bounds of each time delay before the system becomes unstable. Each delay should have a bound before it causes instability in the control loop, and then an analysis can be made to determine the magnitude each delay can have before the system fails [10]. Once achieved, the control can be placed on hardware systems to examine the validity in a real-time situation.

REFERENCES