MIDDLEWARE AND ARCHITECTURE FOR ADVANCED APPLICATIONS OF CYBER-PHYSICAL SYSTEMS

A Dissertation

by

WOO HYUN KO

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	P. R. Kumar
Committee Members,	Suman Chakravorty
	Le Xie
	I-Hong Hou
Head of Department,	Miroslav M. Begovic

December 2017

Major Subject: Electrical Engineering

ABSTRACT

In this thesis, we address issues related to middleware, architecture and applications of cyber-physical systems. The first problem we address is the cross-layer design of cyber-physical systems to cope with interactions between the cyber layer and the physical layer in a dynamic environment. We propose a bi-directional middleware that allows the optimal utilization of the common resources for the benefit of either or both the layers in order to obtain overall system performance. The case study of network connectivity preservation in a vehicular formation illustrates how this approach can be applied to a particular situation where the network connectivity drives the application layer.

Next we address another aspect of cross-layer impact: the problem that arises when network performance, in this case delay performance, affects control system performance. We propose a two-pronged approach involving a flexible adaptive model identification algorithm with outlier rejection, which in turn uses an adaptive system model to detect and reject outliers, thus shielding the estimation algorithm and thereby improving reliability. We experimentally demonstrate that the outlier rejection approach which intercepts and filters the data, combined with simultaneous model adaptation, can result in improved performance of Model Predictive Control in the vehicular testbed.

Then we turn to two advanced applications of cyber-physical systems. First, we address the problem of security of cyber-physical systems. We consider the context of an intelligent transportation system in which a malicious sensor node manipulates the position data of one of the autonomous cars to deviate from a safe trajectory and collide with other cars. In order to secure the safety of such systems where sensor measurements

are compromised, we employ the procedure of "dynamic watermarking". This procedure enables an honest node in the control loop to detect the existence of a malicious node within the feedback loop. We demonstrate in the testbed that dynamic watermarking can indeed protect cars against collisions even in the presence of sensor attacks.

The second application of cyber-physical systems that we consider is cyber-manufacturing which is an origami-type laser-based custom manufacturing machine employing folding and cutting of sheet material to manufacture 3D objects. We have developed such a system for use in a laser-based autonomous custom manufacturing machine equipped with real-time sensing and control. The basic elements in the architecture are a laser processing machine, a sensing system to estimate the state of the workpiece, a control system determining control inputs for a laser system based on the estimated data, a robotic arm manipulating the workpiece in the work space, and middleware supporting the communication among the systems. We demonstrate automated 3D laser cutting and bending to fabricate a 3D product as an experimental result.

Lastly, we address the problem of traffic management of an unmanned aerial system. In an effort to improve the performance of the traffic management for unmanned aircrafts, we propose a probability-based collision resolution algorithm. The proposed algorithm analyzes the planned trajectories to calculate their collision probabilities, and modifies individual drone starting times to reduce the probability of collision, while attempting to preserve high performance. Our simulation results demonstrate that the proposed algorithm improves the performance of the drone traffic management by guaranteeing high safety with low modification of the starting times.

DEDICATION

To my parents, my parents-in-law, my lovely wife, and my precious Chris and Chloe

ACKNOWLEDGEMENTS

Most of all, I would like to thank to my advisor, Dr. Kumar. He has always shown his generosity, given great advices for my research, supported whatever I wanted to do, and encouraged me in all my research works. Thanks to his ongoing support, I could finish my dissertation research. I always respect his great enthusiasm for teaching and researching as well as his brilliant achievements. I am grateful to my committee members, Dr. Chakravorty, Dr. Xie, and Dr. Hou, for their valuable guidance and support throughout the course of my dissertation research.

I would also like to thank to my friends and the department staffs for making my time at Texas A&M University a great experience. I really enjoyed playing basketball and soccer with my friends every Sunday. It totally relieved my stress no matter how bad my situation was. Thanks to Dr. Yoon, his counseling enabled me to stop wandering aimlessly through the life of an international student. Carolyn and Vickie were always good problem solvers for me whenever I was in trouble. Their invaluable help made my study life go well smoothly.

Finally, thanks to my mother and father for their encouragement. They always trusted me and supported my choice in career. To my wife, I cannot thank enough for her patience and love. She has always been there for me and supported me without any complaint although it was a really harsh and long journey for her to take care of our family during my dissertation research. To my son and daughter, they have been my precious joy all the time. Their lovely smiles and hugs always made me very happy.

For the whole thing, thank God.

CONTRIBUTORS AND FUNDING SOURCES

This work was supported by a dissertation committee consisting of Professor P. R. Kumar, Professor Le Xie, Professor I-Hong Hou of the Department of Electrical and Computer Engineering and Professor Suman Chakravorty of the Department of Aerospace Engineering.

Graduate study was supported by a National Science Foundation (NSF) under Contracts Nos. CNS-1302182, ECCS-1547075 and CNS-1646449, and Science & Technology Center Grant CCF-0939370, the U.S. Army Research Office under Contract No. W911NF-15-1-0279, and National Priorities Research Program (NPRP) grants 6-784-2-329 and 8-1531-2-651 from the Qatar National Research Fund, a member of Qatar Foundation.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

We address the problem of cross-layer design and operation of cyber-physical systems (CPSs) which consist of a cyber layer, comprised of computation and networking, and a physical layer, comprised of sensors, actuators and the physical plant. We propose middleware and architecture that make feasible cross-layer design and operation spanning both the physical and cyber layers. Our design treats the application layer situated above the middleware, and the operating system situated below the middleware, in a symmetric fashion, allowing bi-directional interactions. It thereby makes possible, for example, systems involving either "control for communication" or "communication for control". An example of the former is when the positions of drones in a fleet need to controlled so that they give rise to a connected wireless network, while an example of the latter is when the routing table entries of the communication network need to be modified so that they can support a particular fleet of drones engaged in a surveillance application. We implement and demonstrate the proposed middleware design in a laboratory cyber-physical systems testbed through two experiments. The first features control of two drones to maintain connectivity. The second features removal of outliers caused by unreliability of packet deliveries in the communication network layer.

We also consider three advanced applications of CPS. First we consider the problem of security of CPS. We extend a pro-active cyber-security measure called Dynamic Watermarking, a recently advanced cybersecurity measure [34]-[37][42][43], to detect and stall attacks on the position sensors of an automated transportation system. We show that the middleware design supports the application of dynamic watermarking, to

detect attacks on sensors or actuators and thereby ensure security and cyber-safety of the overall system. We demonstrate the success of this method in a laboratory version of an autonomous transportation system.

Next we consider the problem of cyber-manufacturing. We demonstrate an origami-based method that employs a laser to perform automated folding and bending of sheet material to manufacture 3D objects. This has been proposed as a quicker alternative to time consuming 3D printing or additive manufacturing.

Last, we consider the problem of traffic management of an unmanned aerial system (UAS) of drones, envisaged to operate 200ft-500ft above ground level in the lower portion of the Class G airspace [47]. We consider the problem of detecting likely collisions in flight paths of vehicles and how to mitigate them. We design a probability-based collision detection and resolution algorithm for autonomous systems of drone. This plays a key role in the off-line flight planning protocol in a proposed complete design of both communication and control layers of an unmanned aerial system traffic management (UTM) system, whose primary goal us ensuring system-wide safety from collisions.

We now elaborate on the above.

Cyber-physical systems consist of a cyber layer comprised of a computational and a communication network, and a control system featuring sensors and actuators interfaced with a physical plant. Typically, the operation of the cyber-layer, for example, the communication network, does not require any awareness of the state of the control system. Conversely, typically, the control policy is determined without regard to the state of the communication network. Their designs are usually separated at the very outset for the

convenience of development. Such separation simplifies the design process, since it enables each to focus on the issues in its own domain, under some assumptions about the other domain. In fact, one important objective of middleware is indeed to hide the details of the operating system from applications. This can however result in ignorance of the operating system state, and, in many applications, such separation can severely restrict the overall system from attaining its objective, or lead to a serious limitation on its performance. One example is to control the mobility of a formation of unmanned aerial vehicles (UAVs) in such a manner that they always form a wirelessly connected unit, regardless of any obstacles or the propagation environment. In such a situation, it is imperative for the position control system to be aware of whether the communication network's routing table indicates connectivity to all the nodes involved. In order to deal with such issues, we propose a cross-layer design supporting more general interactions between networking and control than is possible through a separated design.

We propose a cross layer design method over "bi-directional" middleware. By "bi-directional middleware" we mean middleware that supports and facilitates interactions between the cyber layer, comprised of computation and networking, and the control layer. Through the proposed bi-directional middleware, the networking layer can not only obtain appropriate information about the physical layer to manage networking resources in an efficient way, but the reverse is also possible. Such information can be used, for example, to constrain the control of the physical layer to prevent network disconnection caused by an undesirable state of the system. Likewise, the physical layer can also access appropriate

information about the networking layer and make changes in the networking layer such as changing the network's topology as is suitable for desired objective of the physical plant.

We demonstrate the proposed cross-layer design over bi-directional middleware through experimentation with the developed middleware in two scenarios featuring the interaction of the networking and control layers. In the first scenario, we consider the problem of the preservation of network connectivity during formation control of multiple vehicles in our cyber-physical systems testbed.

In the second experiment, we consider an application where errors caused by packet delivery timing have to be corrected for in the application layer. Generally, in cyber-physical systems, errors in one domain can spill over into the other domain. For example, network delay jitter can lead to impaired control system performance. We demonstrate the capability of the middleware to address such situations through a scenario where the application layer needs to compensate for unreliability in packet delivery errors. Recently, there has been a growing trend to use model predictive control (MPC) in many industries. Proportional integral derivative (PID) control has been popular in industry for over a half century due to its easy design, quick deployment and acceptable performance. However, as industrial systems have become larger in scale and more complex in configuration, PID control is in many applications not well suited to stabilize a large-scale cyber-physical system. In many applications, MPC is substituting for PID control because it guarantees better performance through the exploitation of the availability of a system model. However, if there are some uncertainties in the dynamic system model, the control performance of the MPC can be seriously threatened. To address the lack of knowledge

of the system model with the desired precision, *adaptive* MPC employs observation data to reduce model mismatch with the goal of obtaining more precise system model and thereby better control performance. However, a large-scale CPS has a somewhat different and non-traditional aspect that can affect its control performance. The perception information of the CPS can be impaired not only by normal state measurement noise, but also by various networking errors such as packet loss and packet delay. One example is timing errors in the measurement data which are used to estimate the state of the system. Such estimates can be especially harmful if they contain outliers that degrade the model adaptation involved in adaptive MPC, and so, deteriorate the control performance.

Some researchers [21] have suggested removing outliers with 5th order mean filters, but that does not adequately eliminate outliers when they are frequent and consecutive. Other researchers [22] have proposed using a system model to reject outliers and have established improved effectiveness for outlier rejection. However, model mismatch is common, and so this approach may not be sufficient to get rid of all outliers in a real system. To alleviate this problem, we propose a method consisting of flexible model adaptation with outlier rejection to simultaneously remove uncertainties in a system model and outliers in estimates from observation data. The proposed algorithm has two phases. The first phase is outlier rejection which removes outliers from the measurements based on estimation using a system model along with the previous control commands, and then smoothens the filtered measurements by using a maximum-minimum exclusive mean (MMEM) filter. The second phase is model adaptation which updates the parameters of the system model by using a least-squares method with the outlier-free data. The two

phases together improve the precision of the system model as well as the performance of the outlier rejection. The experimental results show that the proposed method indeed improves the tracking performance of a battery-powered car which has a mismatch between the battery power model for its MPC and the real battery power.

We next turn to three advanced applications of CPS, one in autonomous ground transportation, one in cyber-manufacturing, and one in flight planning of an unmanned aerial transportation system. In one transportation application we consider the problem of security and safety of an automated transportation system. We also illustrate the capability of the proposed middleware to implement the two suggested real-time solutions.

In the first application we consider the threat posed to automated cars by malicious hacking of sensors in the system. There have been several reported cases of hacking autonomous vehicles [29]-[33] emphasizing that such concerns are not groundless. More generally, there has been increasing concern about the threats to physical plants from cyberattacks, as traditional systems evolve to networked cyber-physical systems. One technique that appears to be hold at least some promise of being a general purpose method is "dynamic watermarking" [34]-[37][42][43]. This is an active defense method suggested for CPS security. It attempts to detect the existence of malicious nodes in a closed loop that includes an honest node. The honest node adds a random signal as a watermark onto its output data, and inspects its input data to see if it contains the watermark or not. The absence of the watermark is evidence that one of the nodes in it's the closed loop has manipulated the real data.

The present work extends previous studies [37][43] of dynamic watermarking to nonlinear systems of the type arising in vehicular models. We implement the resulting dynamic watermarking procedure in an intelligent transportation system and demonstrate how it successfully detects a malicious sensor node, thereby securing the safety of the vehicles by preventing collisions.

The second application we consider is cyber-manufacturing. Advanced technologies have led to many innovations in the manufacturing industry, such as productivity enhancement by automation technology, facility enlargement by information communication technology, and quality improvement and product diversity by manufacturing process technology [48][49][50]. Employing these innovations, the manufacturing industry is proceeding towards producing customized and high quality products at low costs.

3D printing technology is emerging to lead the manufacturing industry. However, additive-type 3D printing has some shortcomings like long process times, unhealthy powder additives, and limit to product sizes. In order to resolve these issues, a laser origami technology has been introduced [51]. It rapidly manufactures customized 3D creatures by using a laser to cut, bend or fold sheet material. However, manually operated bending processes require a human operator to perform and monitor the whole process. As the switching between cutting and bending processes is performed by a human operator, he or she can be exposed to hazards caused by the laser. Moreover, it takes a significant amount of time to determine the proper intensity of the laser beam and the feed rate of a laser head, often through a trial and error process. Also, the experience level of

the operator can affect the quality of the product due to poor choice of intensity or feed rate. It can therefore be difficult to guarantee the quality of the product with manual bending.

In contrast, our proposed feedback-based automatic laser fabricating system can not only assure the safety of the human operator, but it can also reliably and repeatedly reproduce high-quality products. Even more, considering the scalability of the automatic system, massive and customized production with high-quality is enabled.

We implement an automated laser origami system for cyber manufacturing as an advanced application of CPS. It consists of a sensing system with cameras monitoring the status of the manufacturing process, a control system that controls both the feed rate of the laser head as well as the intensity of a laser beam, based on feedback information, to operate the cutting and bending processes automatically. While the network enables the laser, sensing and control systems to communicate with one another, the middleware enables component-based development which offers quick deployment, easy extension, and ready maintenance. We validate the proposed system by experimenting with laser bending in the laser origami system.

Last, we address the flight planning portion of an unmanned aerial system. We consider the improvement of the air traffic throughput of the system while maintaining collision free trajectories. Recently, automated drone systems have generated wide interest in various applications, and a drone industry appears to be imminent. Its potential rapid emergence is bringing a demand for the establishment of standards for an unmanned aircraft system and the development of a traffic management system. To keep pace with

the demands, many private companies, such as Google, Amazon, CNN, etc., and national agencies, such as National Aeronautics and Space Administration (NASA) and Federal Aviation Administration (FAA), are collaborating to prepare for such a future of populated drones in the airspace over cities [63][64][65][66].

An unmanned aerial system traffic management is needed for such a drone system, in contrast to the traditional manned aircraft traffic management. As the number of manned aircraft under management usually has a limit, it is possible to manage them based on voice supervision by humans. In contrast, the number of unmanned aircraft is expected to be so large, and with such a high density of drones flying in close proximity in irregular trajectories, that it will outstrip the capability of a manned air traffic system. An automated administration appears to be inevitably needed for an aerial system traffic management of drones. However, it will be necessary to endow drones with safety protocols, since an unmanned aircraft lacks the ability to handle various situations in a dynamic environment as human pilots do.

An automated protocol-based air traffic management for unmanned systems should consider safety and throughput. When a great number of drones fly in a dynamic environment with uncertainties, an online protocol should guarantee the safety of the drones by preventing any collisions in real time. Also needed is an off-line protocol to improve the throughput of the airspace traffic system by deconflicting their trajectories at the initial flight planning stage.

We consider the design of such an off-line protocol at a central server to approve the trajectories of drones to reduce the traffic congestion in the airspace. In the envisaged

system, [67] the drones of one company will have no information on the trajectories of other drones operated by other companies. When two drones confront one another due to the intersection of their trajectories, they detour their planned paths or make an emergency landing to avoid a collision, according to their online protocol [60]. As a result, their actions taken for real time collision avoidance in the populated airspace can lower the throughput of the entire air traffic.

There have been some studies about multi-agent collision avoidance which can be applied to solve the problem of collision detection and resolution among drones' trajectories. Park et. al. suggested a collision map method, which identifies a collision segment occupied by other agents along the agent's path to find collision-free trajectories for a multi-agent system [61][62]. They assign a priority to each agent and resolve collisions among their trajectories by delaying the starting time of the lower-priority agent. However, the priority-based algorithm has the disadvantage that the high density of the population of agents greatly increases delays of the starting time of the lowest-priority agent, and is therefore unfair to such agents. In addition, it can thereby also lower the number of agents concurrently starting, and consequently reduce the throughput.

We propose a probability-based collision detection and resolution algorithm not only to reduce collision probabilities among the trajectories of drones, but also improve the throughput of the air traffic for an UTM. Due to uncertainties caused by wind, etc., the future position of a drone will generally be uncertain. We model the position of a drone by a Gaussian distribution. The proposed algorithm calculates collision probabilities among drones' trajectories and modifies their starting times to reduce the collision

probabilities. Instead of assigning a priority, a round-robin method is used to adjust each trajectory's starting time to reduce its collision probability iteratively so that the total deviation of the modified starting times by the proposed algorithm becomes much less than that of the priority-based collision resolution algorithm.

We conduct a simulation study of the proposed algorithm. In the simulation, we randomly generate multiple 3-dimensional trajectories and resolve their potential collisions by using the proposed probability-based algorithm and the nominal priority-based algorithm, respectively. We demonstrate that the proposed algorithm effectively reduces the number of collisions with higher probability and achieves smaller modification in the starting times of the trajectories than the priority-based algorithm.

In Section 2 we describe the middleware and its architecture, and the testbed of the Cyber-Physical Systems lab. In Sections 3 and 4 we propose the cross-layer design approach with bi-directional middleware and address the interaction between the networking layer and the physical layer, and uncertainties of the system model in the physical layer and outliers caused by communication networking in the networking layer. In Section 5 we describe the implementation of Dynamic Watermarking in an intelligent transportation system, as well as an automated 3D laser origami system for cyber-manufacturing. In Section 6 we describe probability-based collision detection and resolution for unmanned aerial system traffic management.

2. SOFTWARE AND HARDWARE PLATFORMS

We begin by describing the middleware of interest in cyber-physical systems.

2.1. Middleware: Etherware

Through middleware we can divide a cyber-physical system into two layers: the cyber layer containing computation and networking, and the physical layer containing the sensors, actuators and the physical plant. Network connections among nodes, sharing information, and message delivery, occur in the cyber layer, while actuation, sensing and controlling occur in the physical layer in interaction with the environment.

Etherware is such a middleware for networked control systems [1]-[4]. It makes connections and manages communication among computing nodes in the network. It also allows prioritization, and thus the incorporation of real-time considerations. At the application level, it supports a component-based architecture. Typically, the application designer simply develops components containing the application algorithms. These components are then executed by the Etherware Kernel, which completely manages the exchanges of all messages to support the required communication among components.

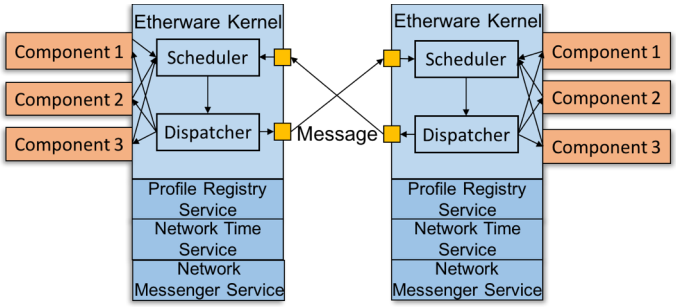


Figure 2.1: Architecture of Etherware.

Figure 2.1 shows the Etherware architecture. The messages created by components are handled by the scheduler of the kernel and they are processed in FIFO order by the dispatcher of the kernel. The message communication among components is managed by three Etherware services, a Network Message Service, a Network Time Service, and a Profile Registry Service. The Network Message Service maintains the connections, and the sending and receiving of messages among components. The Network Time Service synchronizes the clocks at different computing nodes by calculating the message transit time. It allows a component to recognize the timestamp of the received message by a local clock. The Profile Registry Service provides for semantic naming of the component addresses. It allows components to use a semantic name for the message receiver for sending messages, based on which the Profile Registry Service determines the IP address of the message receiver by looking up a profile table that is a map between IP addresses and semantic names.

2.2. Testbed: Cyber-Physical Systems Lab

The testbed over which we conduct several of our experiments is a multiple vehicle control system housed in the Cyber-Physical Systems Laboratory at Texas A&M University. Figure 2.2 provides an overview of the testbed of the Cyber-physical systems lab. The work space measures 5m x 4m x 2m. The vision system consists of 10 Vicon Boneta 10 cameras and two desktop computers. One of the vision processing computers determines the position and orientation of vehicles with 100 Hz frequency, and the other vision processing computer gathers the information and distributes it to other components. A high level controller computer generates trajectory data. It functions as a supervisor.

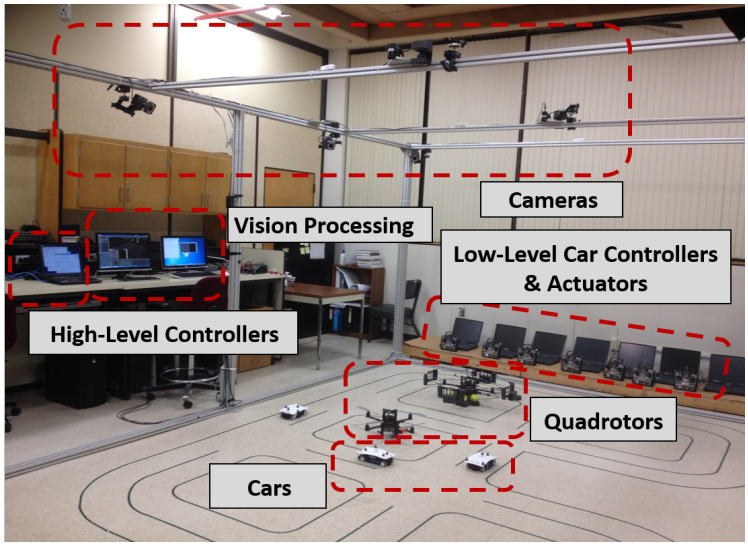


Figure 2.2: Overview of the testbed in the Cyber-Physical Systems Lab.

There are several other computers that act as low level controllers and actuators to control cars. Such a low level controller determines the control commands, such as the next speed and steering angle in the next sampling period, so as to follow a trajectory. An actuator moves a car by transmitting an RF signal containing the commands. Individual cars are controlled by individual laptops which each have a low level controller and an actuator with a transmitter. In addition, there are several other vehicles such as remote controlled cars and Pelican quadrotors. The quadrotors have an Atom board on which Etherware runs. All computing nodes are connected by a network through Etherware.

3. CROSS-LAYER DESIGN FOR CYBER-PHYSICAL SYSTEMS OF COORDINATED NETWORKED VEHICLES OVER BI-DIRECTIONAL MIDDLEWARE*

Networked control systems consist of (i) a communication network connecting sensors, controllers and actuators, and (ii) a control system closing the sensor-plant-actuator loop around a physical plant. Typically, the operation of the communication network, for example the routing policy, does not take the state of the plant in the control system into account. Similarly, typically, the control policy specifying inputs applied to the plant does not take the state of the communication network into account. The designs are isolated from each other at runtime. However, in many applications, one can obtain a better overall system by in fact taking the states of both into account in the design of both of them. Some applications would even be infeasible if such capability is not available.

The separation of the designs of control and communication policies is intended to enable each to concentrate on the problems in its own domain. However, their conservative or even idealized assumptions about the other domain can deteriorate the total system performance when they are implemented together into a real system. Therefore, it is necessary to allow interactions between the control layer and the communication layer to improve system performance. This is called, variously, co-design or cross-layer design or joint design.

---

* Reprinted with permission from "Cross-layer design for cyber-physical systems of coordinated networked vehicles over bi-directional middleware" by Woo-Hyun Ko and P. R. Kumar, 2016. *IEEE ACC*, pp. 6459-6464, Copyright 2016 by IEEE

Motivated by this, there has been recent research in the area of cross-layer design. Trivellato et al [5] suggest a general framework for optimal cross-layer design for signal quantization and network resource allocation. Under the assumption that controllers and actuators use a TCP-like protocol, they solve the problem of optimal control based on packet drops and signal quantization for a stable system in a target state and an unstable system with large transmission bandwidths. Branicky et al [6] study the arbitration of networking scheduling and bandwidth when a set of networked control systems are connected in a network, and formulate the optimal scheduling problem under rate-monotonic schedulability constraints and stability constraints. Typically, as above, there has mostly been analysis of the effect of networking on control performance.

However, because the interactions between two layers are not one-directional but bi-directional, it is necessary to also take account of the effects of control on networking. For example, in a wirelessly networked control system, the physical environment of the nodes can influence the networking performance of the system. If network connectivity is lost, some part of the system may get out of control so that the total system is unable to perform its task. For preservation of network connectivity, Michael et al [7] have considered a distributed connectivity control algorithm based on the relative distances between robots. However, physical distance between robots alone cannot guarantee network connectivity without radio received signal strength information (RSSI), because radio interferences or fading due to obstacles can occur in real dynamic environments.

In this section, we suggest a cross-layer design approach for cyber-physical systems that employs a “bi-directional” middleware to address the problems relating to

the interaction between the networking layer and the control layer. This bi-directional middleware is built on Etherware, which supports component-based development for composability and transparency in cyber-physical systems. In particular, this bi-directional middleware makes it possible to address the problem of the interactions caused by the effects of a control law on networking performance as well as networking protocols on control. Our goal is to show that the suggested design greatly facilitates the building of systems for which information needs to be exchanged between layers.

We illustrate the improvements for implementability and compatibility that can potentially be realized through such a cross-layer design implemented via bi-directional middleware with an experiment conducted in the testbed of our cyber-physical systems lab. In the experiment we manage the interaction between the networking layer in the cyber domain and the physical layer to maintain network connectivity between a car and a quadrotor.

The cross-layer design for cyber-physical systems is described in Section 3.3. We propose bi-directional cross-layer design for cyber-physical systems in Section 3.4. Section 3.5 contains the results of the experiment.

3.1. Cross-Layer Design for Cyber-Physical Systems

While the separation of the layers, as shown in Figure 3.1 (a), decreases the complexity of the system design, it can cause critical performance problems in the phase of system integration, because of either conservative or idealized assumptions made about the other layer.

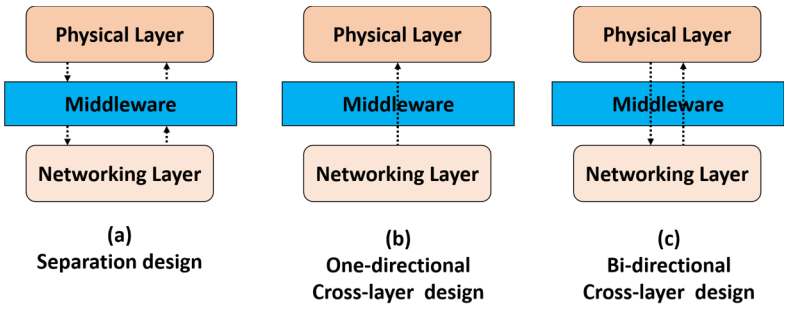


Figure 3.1: Cyber-physical system designs with a separation-based middleware, a one-directional middleware, and a bi-directional middleware.

Consider the problem of controlling a formation of multiple quadrotors communicating with each other over a multi-hop wireless network, where their positions are controlled under the idealized assumption that network connections are ensured when their relative distances are within a static network range. However, if their wireless signals experience fading due to external objects in a dynamic environment, then network connectivity between quadrotors cannot be guaranteed even when they are close enough to be within static network range. One could pursue a "conservative" alternative where the quadrotors are placed very close to each other to ensure network connectivity. This can sacrifice mission performance, besides being dangerously close. This shows how idealized or conservative assumptions in the separated design can cause loss of performance or even failure in a practical situation.

So motivated, a co-design approach has been advocated to solve the problems caused by the separate design of the cyber and physical systems. Many researchers have suggested designing an optimal control system, taking into consideration practical networking problems such as packet losses, delays and limited bandwidth instead of idealized assumptions, in order to obtain better control performance [8,9]. To illustrate

this, a control system could use received signal strength indications (RSSI) to preserve network connectivity among quadrotors, by controlling the formation. However, while such a one-directional cross layer design, as shown in Figure 3.1 (b), can optimize the performance of the layer in the control domain, it cannot optimize the performance of the entire system when there is a conflict for common resources between networking and control. For example, if the formation of three quadrotors should transform from a tringle into a straight line to go through a narrow pathway, the communication between the first quadrotor and the last one can be poor so that it deteriorates the control performance. In this example, a one-directional cross layer design which manages only the control performance over the network cannot secure the desired mission performance of the entire system.

3.2. Bi-directional Middleware for Cross-Layer Design of Cyber-Physical Systems

We propose a new bi-directional middleware that allows true cross-layer design, making it feasible for layers to not only exchange information but also to interact with each other, in order to obtain a global optimum. The bi-directional middleware, as shown in Figure 3.1 (c), plays the role of delivering data and implementing requirements of either layer in the other layer to arbitrate common resource conflicts between them.

Figure 3.2 shows the middleware-centered layers in the bi-directional middleware. Our bi-directional middleware is a bridge to link the layers. The middleware has access to low level resources in the operating system and the network, as shown in Figure 3.2, and exposes the networking layer, which was earlier hidden under the middleware, to the physical layer. Conversely, it exposes the physical layer to the networking layer.

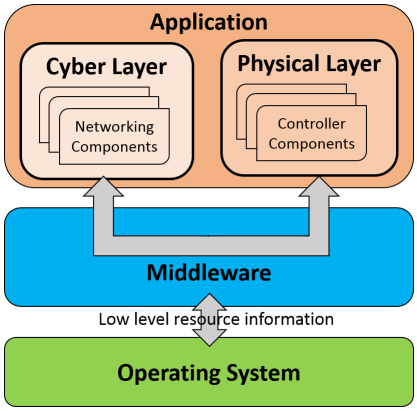


Figure 3.2: The middleware-centered layers.

Several application components such as algorithms, controller and managers in the networking layer and the physical layer are needed to operate a mission. The middleware allows application components to access low level resource information, such as the routing table, RSSIs, the CPU load, and the RAM utilization, provided by an operating system. Similarly, physical information such as position can be accessed by the networking layer as needed. The middleware allows the networking components and the controller components to communicate each other. They are enabled both to share information with each other as well as be executed by it. Thereby the middleware manages the information sharing and interactions between the cyber and physical layers, and enables cross-layer design of cyber-physical systems.

For illustration, we now discuss some of these types of interactions in the context of the quadrotor system above. There can exist conflicts for spatial resources between wireless network connectivity in the cyber layer and formation control in the physical layer – the relative distances among quadrotors should be close enough to maintain their wireless network connectivity while staying far enough apart to safely complete the

mission. To elaborate, while maintaining a formation of vehicles, all the relative RSSIs among them should exceed the needed strength for a stable link for communication, in order to preserve the network connectivity. However, unexpected situations in a dynamic environment such as uneven terrain, moving obstacles, or wireless signal interferences, can make some of vehicles detour as their network signal strengths weaken. For sustainability of the mission, both formation controllability and network connectivity should be preserved all the time. Therefore, coordinated control is needed that can satisfy the requirements of both the cyber layer and the physical layer.

The bi-directional cross-layer design approach provides a solution to the problem of the conflicts between layers by allowing the consideration of both their requirements simultaneously. The requirements of each layer can be used to control the utilization of the resources in the other layer to improve the performance of the entire system. For example, the control system over a network can ask for more bandwidth for communication between some critical vehicles which are the most important vehicles to complete a mission. The network controller can then attempt to meet the additional requirement by allocating more networking resources for these links by, for example, changing routing tables and thus the flows over the network. In the other direction, the network controller can request that the distances between a particular vehicle and some nearby vehicles be shortened to enhance their communication rates since that vehicle is serving as an important hub that ensures the connectivity of the whole network. This requirement provokes a change in the formation of the vehicles by the control system planning and executing the motions of the vehicles.

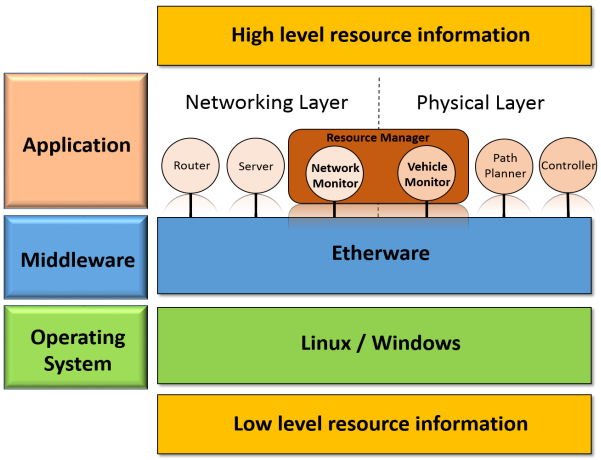


Figure 3.3: Cross-layer design approach with middleware.

In order to handle this interaction between the networking layer and the physical layer, we introduce a Resource Manager module that manages the common resources between the layers. Figure 3.3 shows the cross-layer design approach with middleware over operating system. When there is a conflict of interest between the layers, the Resource Manager module determines how to allocate common resources over space and time.

The suggested bi-directional cross-layer design approach has been implemented in Etherware, which serves as the bridge between the networking layer and the physical layer, and the Resource Manager module handles the resources common to the two. We employ a particularly useful mechanism of Etherware called “Filter”, which allows the system to seamlessly interpose a component into an existing data flow in the network. The Resource Manager module collects information from other components by using the Filter mechanism of Etherware. For example, in order to obtain high level resource information from the control system applications, such as a vehicle's position, the packet drop ratio, and networking traffic status, we employ the “Filter” mechanism, as shown in Figure 3.4. This allows the Resource Manager to determine how to allocate common resources.

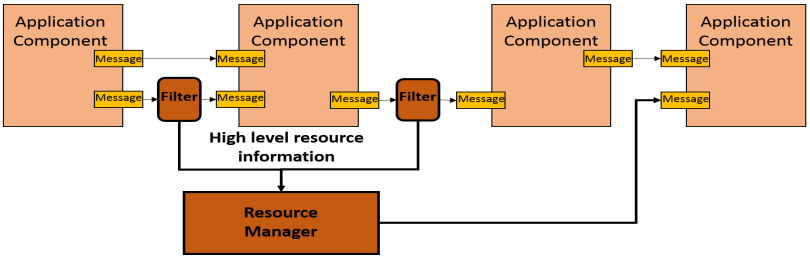


Figure 3.4: The Resource Manager uses the Filter mechanism of Etherware to obtain high level resource information from application components.

The Filter mechanism of Etherware supports Plug-In component incorporation for data provisioning to a new component. Figure 3.4 shows the Plug-In implementation of the Resource Manager with Filter mechanism. The Resource Manager is implemented in an independent module. The Filter modules are placed in the middle of the communication channel between application components. To obtain the messages which the component receives or sends, their destinations are changed into its network address. The Filters intercept and deliver the messages not only to the original destination but also the Resource Manager module. Although an extra latency is incurred, caused by the interception of the messages by the additional independent module, the significant advantage it has is that it is less affected by the algorithm computation and more composable. Moreover, we can use the Filter mechanism to solve the problem of the impact of the network's status, which varies in real time, on control performance, because it allows the system to modify the control algorithms without reconfiguration.

3.3. Experiment: Preserving Network Connectivity

We have described above the methods using middleware for the management of the interaction between the cyber layer and the physical layer. We now describe an

experiment that shows how it applies to the particular situation of interest. The experiment is performed in a multi-vehicle testbed.

The experiment illustrates how to manipulate the interaction between the networking connectivity in the cyber layer and the formation control in the physical layer by using Etherware. In the experiment, a car and a quadrotor follow an oval trajectory. The mission has a distance constraint to preserve network connectivity over a local wireless network. The Vehicle Monitor module keeps watch on their relative distances. If the distance exceeds the constraint, new trajectories are regenerated to preserve the network connectivity between them by the Resource Manager.

For example, during formation control, a Global Path Planner initially employs the resource with the authority to generate a trajectory in the case of a centralized control system, and the Network Monitor of the Resource Manager checks networking status, such as the network topology, to preserve network connectivity. If the change in the network topology is not able to secure the network's connectivity, then Resource Manager restricts the resource of the Global Path Planner so that new trajectories are regenerated to maintain the network connectivity.

In this experiment, it is shown how to preserve network connectivity between a car and a quadrotor during formation control in an unexpected situation. To maintain their formation, the car and quadrotor need to communicate with each other. Over a wireless network, their relative distance needs to be within an effective range to ensure their connectivity. In a dynamic environment, unexpected situations can however cause them to move far away from each other so that the network gets disconnected. The

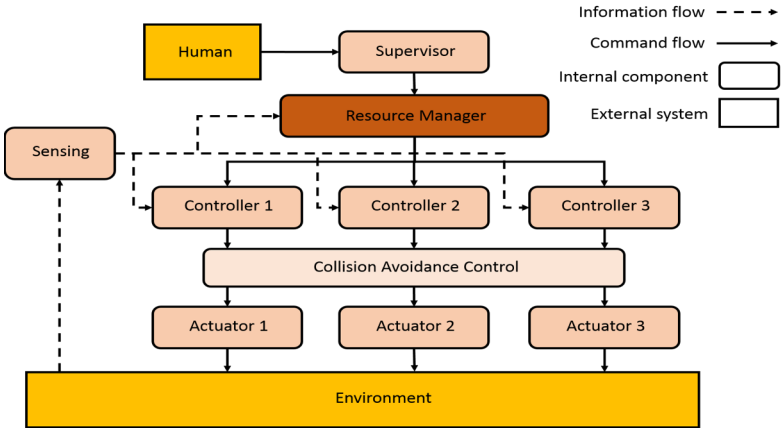


Figure 3.5: Control architecture of a multi-vehicle system that preserves network connectivity.

disconnection makes their formation maintenance difficult since it is important to preserve network connectivity among vehicles in multi-vehicle control when the automated car and the quadrotor plan a new formation.

Figure 3.5 shows the control scheme for multiple vehicles for preserving network connectivity. Given a set of waypoints, the Supervisor plans the trajectories for vehicles and sends them to each vehicle. Each individual vehicle determines its control commands to follow its trajectory based on vision information. The experiment also features a Collision Avoidance Control module that detects collisions between vehicles. When any collision is expected, it regenerates new control commands to avoid the collision [10]. To preserve network connectivity among them, Resource Manager monitors the relative distances between vehicles. When the distances exceed the permitted constraint, it regenerates new trajectories to secure their connectivity.

The experimental system configuration is shown in Figure 3.6. Two cars, one driven by a human, and the other an automated vehicle, and an automated quadrotor, are

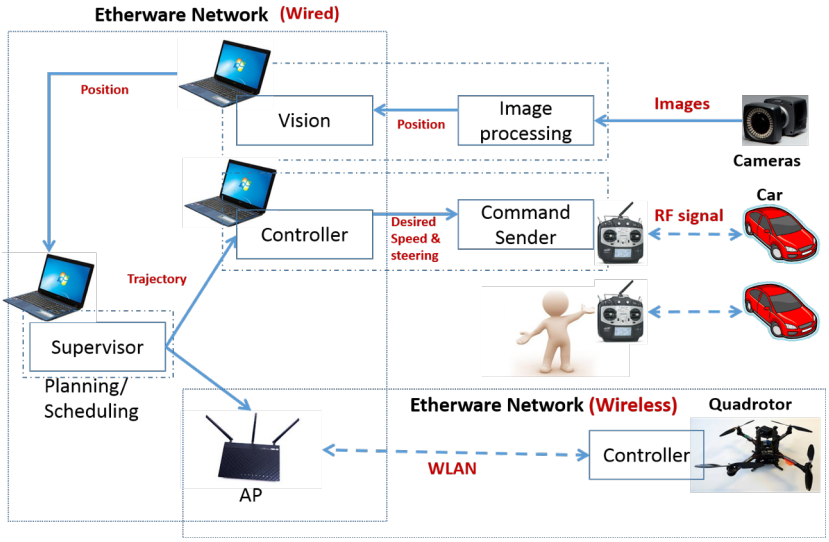


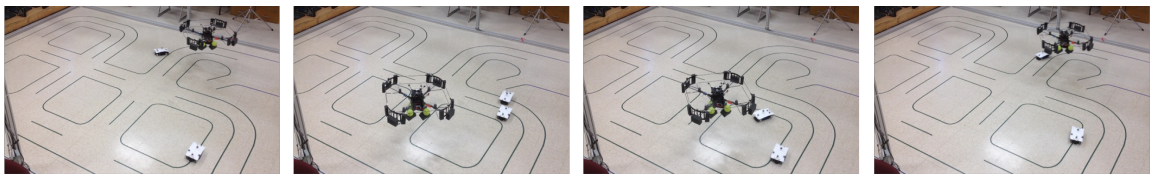
Figure 3.6: Experimental system configuration.

involved in the experiment. The supervisor generates trajectories for the automated car and quadrotor which are connected in the Etherware network. The other car is controlled by a human. The vision system provides global position and orientation of a vehicle.

The experimental scenario is as follows. The automated car and quadrotor receive an oval trajectory and follow it to maintain their formation in the trajectory. There is a distance constraint between them to preserve network connectivity. Their relative distance remains within the distance constraint while they move. A human manually controls another car to interfere and block the moving car. The Collision Avoidance Control module detects the potential collision between two cars and halts the automated car. As the quadrotor continues for a while to follow its originally planned trajectory, it moves further away from the stopped automated car, and their relative distance increases. When the Resource Manager module detects that their separation distance is beyond the distance constraint, it regenerates the trajectory of the quadrotor to maintain the distance remain

within the constraint. When the manually operated car moves under human control and ceases to be an obstacle to the stopped car, the automated car's constraint on movement is released by the collision avoidance system, and it moves to reduce the relative distance to the quadrotor. The quadrotor and the automated car then recover their formation and follow their trajectories sent by the Supervisor after their relative distance decreases below the permitted distance to preserve network connectivity.

Figure 3.7 shows the results of the experiment. There are two cars and a quadrotor in Figure 3.7 (a). The automated car to the left and the quadrotor start by following clockwise an oval trajectory sent by the Supervisor. The automated car moves behind the quadrotor along the trajectory. Figure 3.7 (b) shows that the other car controlled by a human blocks the path of the following car. The quadrotor then waits for the automated car so as to preserve the network connectivity, as per the Resource Manager. When the automated car's path is clear, it moves to the waypoint where the quadrotor is waiting in Figure 3.7 (c). Figure 3.7 (d) shows that they recover their trajectory and at all times avoid exceeding their distance constraint.



(a) An automated car follows an automated quadrotor moving along a clockwise oval trajectory.

(b) The automated car is blocked by human's car and then, the quadrotor is forced to halt to preserve network connectivity.

(c) The path of the automated car is unblocked and it approaches the quadrotor.

(d) The automated car and the quadrotor recover their original trajectory.

Figure 3.7: Experiment for network connectivity between an automated car and a quadrotor.

4. TWO-PRONGED APPROACH INVOLVING A FLEXIBLE ADAPTIVE MODEL IDENTIFICATION ALGORITHM WITH OUTLIER REJECTION ON MIDDLEWARE*

A common methodology in industrial control systems is model predictive control (MPC) [19]. This has also been suggested for use in networked cyber-physical systems, for example in automated vehicular networks [20]. Large scale CPSs will need to provide superior control for multivariable systems, and will need to be deployable with minimum set-up effort and require minimum maintenance. A significant challenge to control performance is that CPSs are subject to errors from a variety of sources in addition to the traditional "measurement noise" in conventional control systems. Packet drops and delays experienced by packets carrying sensor measurement or actuator commands can indirectly cause large outlier types of state estimation errors by causing timing errors. Also, in large scale CPSs, there can be a large number of sensors deployed in remote areas that are not easily accessible. These sensors can suffer from significant calibration errors, or degrade over time, which can again cause large outlier types of errors. More generally, imperfections of system components, such as sensors or actuators, errors in new software updates, or communications layer errors, or timing errors, can all seriously degrade overall system performance.

Motivated by this, we propose a flexible adaptive model identification algorithm with outlier rejection, along with a middleware implementation of it, which can be

*Reprinted with permission from "Outlier rejection for networked control systems based on middleware" by Woo-Hyun Ko and P. R. Kumar, 2017. *IEEE CCNC,* pp. 305-312, Copyright 2017 by IEEE

implemented not just in a clean slate design of a CPS, but can also be inserted as a run-time upgrade into an already running existing control system. The proposed algorithm is implemented on Etherware, which is a middleware for networked control systems that supports component-based development for composability and transparency in cyber-physical systems. We employ the "Filter" mechanism of Etherware which allows a system to modify control algorithms without reconfiguration. It allows an already functioning control loop to be modified, without any need for software reconfiguration, by the interposition of another dynamic component such as a transfer function.

The proposed algorithm consists of both outlier rejection and model adaptation. In the outlier rejection phase, by the Filter mechanism, all measurement data are collected to estimate the states of the system, and outliers are detected based on the deviation of the state estimation from its prediction by using the current system model and the current control command. Then, the outlier-free data are filtered with a Minimum-Maximum Exclusive Mean (MMEM) filter to smooth them. In the model adaptation phase, an adaptive model identification updates the system model by using the least squares method with the filtered data. The two phases are recursively performed to not only reject outliers in the estimated data but also to reduce model mismatch with the system model.

We begin by experimentally demonstrating the presence of such outlier types of errors in a CPS consisting of a vehicular testbed. We show that estimation corrupted by observation and control data containing such outliers, which can be generated by communication network jitter causing timing errors, does degrade the model identification in adaptive Model Predictive Control (MPC), and that such outlier types of errors can be

a key bottleneck that potentially limits performance and reliability. We demonstrate the performance of the overall cyber-physical system for a car control system in a laboratory CPS testbed. First we experimentally demonstrate the significant loss of performance caused by timing errors due to jitter in the communication network. Then we show that the adaptive model identification algorithm with outlier rejection, implemented employing the Filter mechanism of Etherware, improves controller performance by rejecting outliers in the measurement data and reducing the model mismatch with the system model.

The rest of this chapter is organized as follows. After briefly reviewing related works in Section 4.1, the adaptive model identification algorithm with outlier rejection is presented in Section 4.2. In Section 4.3, we describe the mechanisms for algorithm incorporation with Etherware. Section 4.4 contains the results of the experiment.

4.1. Related Work

Gupta, Krishnanand, Chinh and Panda [21] have addressed the issue of outlier detection and data filtering for wireless sensor and actuator networks. The paper adopts the Hodrick Prescott filter (HP filter) to detect outliers in the various sensing data from wireless sensor networks for smart buildings. But, from the experimental results of filtering the data of humidity, temperature and CO2, it concludes that the performance of the moving average 5th order filter is better than the HP filter. Moreover, from the Figures in the paper, it is evident that there still exist outliers in the filtered data of the HP filter as well as the moving average filters. This shows its difficulty of removing outliers completely without a system model. Takahashi, Nonaka and Sekiguchi [22] deal with outliers and occlusion in the localization by using 3D static cameras and introduce a

Moving Horizon Estimation (MHE). The evaluation of the proposed method was performed by comparison of the estimation accuracy with the Extended Kalman Filter. The additional constraints on the difference of position, heading angle and steering angle during one sampling time to be less than some specified values reduce the influence of outliers and occlusion. However, the limits employed in the additional constraints are dependent on a system model.

Since the performance of MPC is influenced by model accuracy, adaptive MPC has been proposed to handle model uncertainties by updating the system model based on the on-line measurement data. In a CPS vehicular context, Chen, Luan and Lee [23] suggest adaptive model predictive control with a linear time-varying prediction model for lane keeping control. They employ real-time on-line system identification using recursive least squares to estimate the tire cornering stiffness of the bicycle model, and minimize a cost function comprising the steering angles and the errors between the target trajectory and predicted trajectory, to obtain the steering angles within the prediction horizon. Their simulation results show the effectiveness of the proposed control by reducing the lateral displacement error. However, they have not considered outliers which can frequently occur in real measurement data and severely degenerate the results of a reasonable model identification procedure [24]. Kim, Fukushima and Sugie [25] present a robust adaptive model predictive control algorithm which consists of an on-line parameter estimation, and an MPC method based on the modified comparison model. However, the proposed estimation algorithm requires additional computing power to obtain the smallest

eigenvalue as compared with the conventional estimation algorithm, which can increase the computation load in the controller.

4.2. Flexible Adaptive Model Identification with Outlier Rejection

The performance of a controller usually relies on state estimates as well as the system model to determine control commands. When an estimate of the state of a system is provided to a controller, that estimate can have outliers. In cyber-physical systems outliers can be caused by jitter in packet delivery over the communication network that results in timing errors, which in turn results in erroneous state estimates of the physical portion of the system. The resulting contamination in such state estimates provided to a controller can significantly deteriorate control performance. In addition, control performance can be influenced by the accuracy of the model which a controller assumes, which can be affected by calibration errors of sensors or actuators. In order to improve controller performance, a system model is usually used to estimate the response of a system to inputs or the effects of disturbances. However, one does not have a perfect model for what is generally a nonlinear, time-varying system corrupted by noise. For example, the rotations per minute of a DC motor is influenced by battery power output, and the battery model for an electric car typically does not exactly match with the real system because the latter involves an unknown nonlinear function of time that is related to the power consumption to move a car.

Such outlier and model mismatch needs to be addressed to obtain satisfactory control performance. Paradoxically, in order to reject outliers from measurements and estimates, a system model can be used. It provides a guideline to determine whether data

are outliers or not. Moreover, a system model can itself be adapted by taking advantage of the estimates. A good system model enables the obtaining of good estimates and vice versa. Therefore, outlier rejection and model adaptation have a mutually complementary relationship. In this section, we focus on how to achieve both accurate estimates and a precise system model simultaneously. For the application of the suggested algorithm, we consider a networked vehicular control system with a model-based controller. A time-varying nonlinear dynamic model with velocity control inputs is used for the vehicular motion. A controller receives observation data from remote sensors through a network determines control commands based on the state prediction with its system model:

$$x_{k+1} = f(x_k, v_k(i_k)), \tag{4.1}$$

where x_k is the state of a vehicle, i_k a control input at time k, f is a kinematic model, and $v_k(\cdot)$ is a time-varying dynamic model nonlinearity for velocity as a function of the control input. We assume that the velocity model $v_k(\cdot)$ varies slowly with time. The suggested algorithm recursively determines the precise $v_k(\cdot)$ at each update time with the outliers which contaminate state estimation.

We illustrate the problem and our solution on a prototypical situation arising in vehicular control of battery powered vehicles. There are two steps in the outlier rejection algorithm. The first step is to detect outliers in velocity estimates using a system model [1]. We consider the typical problem in vehicular control where there is a finite number of control velocities between the minimum velocity and the maximum velocity, sorted so that the lower indices correspond to lower velocities:

$$V = \{(i, v^m(i)) \mid i = 1,2, \dots, M\}, \tag{4.2}$$

where $v^m(i) < v^m(j)$ if $i < j$ and $i, j \in \{1, 2, \ldots, M\}$.

Velocity actuation is typically performed through selection of an "index." However, each index corresponds to only an imprecisely known velocity. Moreover, it changes with time as the battery is used.

We consider the following algorithm for estimation of the velocity model. At first, the current velocity is estimated from the observation data which have position data with timestamps. If the velocity estimate is beyond the valid range, it is regarded as an outlier and replaced with the minimum or maximum value in the valid range. Letting the velocity estimate at time k be v_k^o, if the pre-applied command index is i, we employ the estimate

$$\hat{v}_k^o = \begin{cases} v^m(i+1) & \text{if } v_k^o \geq v^m(i+1) \\ v^m(i-1) & \text{if } v_k^o \leq v^m(i-1) \\ v_k^o & \text{otherwise} \end{cases}. \tag{4.3}$$

The second step is to smooth the estimates with a Minimum-Maximum Exclusive Mean (MMEM) filter which calculates the mean value of *L-2* values among the set O_k in equation (4.4) which contains *L* consecutive values except their minimum and maximum [27]. This reduces the variation of the estimates so that we obtain more precise estimates:

$$O_k = \{\hat{v}_{k-L+1}^o, \hat{v}_{k-L+2}^o, \ldots, \hat{v}_k^o\}, \tag{4.4}$$

$$\bar{\hat{v}}_k^o = \frac{1}{(L-2)}\left(\sum_{i=k-L+1}^{k} \hat{v}_i^o - \max\left(\hat{v}_j^o\right) - \min\left(\hat{v}_j^o\right)\right), \tag{4.5}$$

where $k \geq L$ and $(k - L + 1) \leq j \leq k$.

Next, using the filtered velocity estimates, the Model Adaptation algorithm updates the velocity model of a car according to the battery power output. The initial

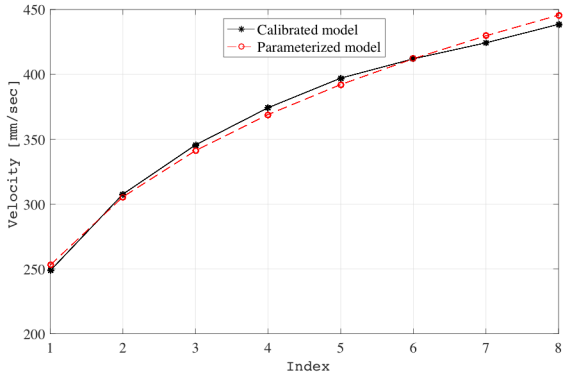


Figure 4.1: The parameterized velocity model.

calibration of the velocity model is usually done offline. Subsequently the velocity model of the car needs to be adapted to the current battery power at runtime.

After collecting enough estimated data points, a Model Adaptation Module finds the velocity model best matching the updated velocities and command indexes by using a least squares method.

The solid line in Figure 4.1, the statistical velocity model showing the relationship between forward indices and the resulting velocities for a particular value of the battery power, is obtained by the calibration procedure. In order to approximate such a nonlinear model, we use an exponential function, which can be linearized by a logarithm, and fitted by a least squares method to determine the best estimated curve from the statistical data. The dotted line in Figure 4.1 shows the nonlinear velocity model parameterized in the equation (4.6).

The following procedure shows how to obtain the approximation from the observation data by using a least squares method. We employ a particular parameterized

nonlinear velocity model for forward indices describing the relation between command indexes and velocities:

$$v(i) = \alpha \cdot i^{\beta}, \tag{4.6}$$

where v is a velocity, i is a command index, and α and β are parameters. We can obtain a linear equation by taking logarithms:

$$d = a + \beta \cdot c\,, \tag{4.7}$$

where $d = \log(v(i))$, $c = \log(i)$ and $a = \log(\alpha)$.

After collecting K pairs of observations and command index,

$$S = \left\{\left(c_k\,,\, d_k\right)\middle| c_k = \log i_k\,,\; d_k = \log\, \hat{v}_k^o,\;\; k = 1,\; 2,\;\; \ldots,\; K\right\}, \tag{4.8}$$

we can rewrite (4.7) in a linear regressor form suitable for forming least squares estimates:

$$D = C \cdot \begin{bmatrix} a \\ \beta \end{bmatrix}, \tag{4.9}$$

where $C = \begin{bmatrix} 1 & c_1 \\ 1 & c_2 \\ \vdots & \vdots \\ 1 & c_K \end{bmatrix}$ and $D = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_K \end{bmatrix}$.

The least squares parameters estimates is then

$$\begin{bmatrix} a^* \\ \beta^* \end{bmatrix} = (C^T C)^{-1}(C^T D)\,. \tag{4.10}$$

Finally, we can obtain the estimate of the velocity model for each command index i as

$$V' = \left\{\left(i, v^m(i)\right)\middle|\, v^m(i) = \alpha^* \cdot i^{\beta^*}, i = 1,2, \ldots, M\right\}, \tag{4.11}$$

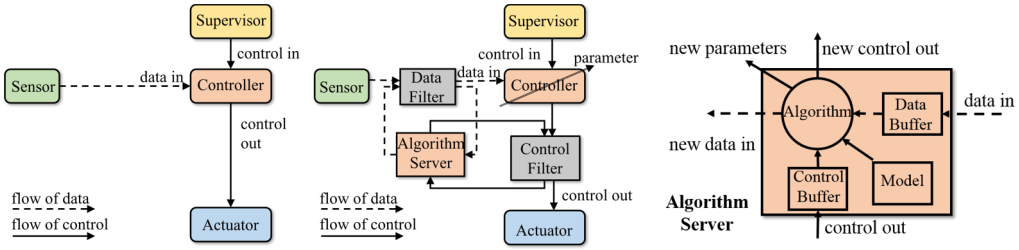
where $\alpha^* = \exp(a^*)$.

This estimate is updated periodically to adapt to the change in the battery state. As a result, the velocity estimates filtered by outlier rejection render the estimation of the velocity model more accurate, while the accurate velocity model in turn allows the algorithm to reject outliers in the estimates obtained from observation precisely, thereby improving the performance of the model-based control system.

4.3. Algorithm Incorporation with Etherware

Large-scale functional systems are composed through the integration of various devices. It is often required to add or replace components to improve system performance or guarantee safety. The specific issue we consider in this paper is the incorporation of advanced algorithms for removing noise and outliers in raw data from sensors for higher performance control. However, traditional system architectures cannot support flexible configuration to incrementally incorporate new algorithms. They incur risk whenever changing system flows and components. We therefore exploit the support provided by Etherware middleware for runtime upgrade to allow system reconfiguration.

We employ the Filter mechanism provided by Etherware. Such a Filter of Etherware can be placed in the middle of the communication stream between arbitrary components without system reconfiguration. It intercepts the messages transferred through the stream and delivers them to the other component. Thereby it can change a message flow in the message communication to support flexible configuration. Figure 4.2 (a) shows the general message flows from the viewpoint of a controller. Data-in and Control-in messages are the inputs to the controller component. A Data-in message that comes from sensors contains the information about the sensor measurement, or



(a) Basic control scheme (b) Incorporation scheme (c) Algorithm server architecture

Figure 4.2: Scheme to incorporate a new algorithm through the Etherware Filter mechanism.

information about the environment, or it may even contain information about the status of the sensor device itself. A Control-in message that comes from a higher level component such as a supervisor can contain information about a reference input or a trajectory command or a mission command. Based on the inputs, the controller component determines a Control-out message to control a device.

Figure 4.2 (b) illustrates the implementation of an algorithm by employing the Filter mechanism of Etherware. A Data Filter is placed between the Sensor component and the Controller component. It intercepts the Data-in message and sends it to an Algorithm Server. In the same way, a Control Filter between the Controller component and the Actuator component intercepts the Control-out message and sends it to the algorithm server. A Filter can also allow a message to go to a destination component, if the message is not supposed to be updated by an Algorithm Server. Moreover, a Filter and an Algorithm Server can be optimally located as appropriate, based on the computational power of the device and the network bandwidth. The new Data-in and Control-out

messages generated by the Algorithm Server are then automatically sent to the Controller component and the Actuator component through each Filter, respectively.

Figure 4.2 (c) depicts the scheme for the Algorithm Server. The inputs, which are Data-in and Control-out messages, are stored in each buffer, and the Model contains device information such as kinematics or dynamics. The algorithm is computed based on the model with the Data Buffer and the Control Buffer to determine new Data-in, Control-out and a parameter set, as outputs. The Algorithm Server sends the outputs to a Controller component or an Actuator component through each Filter.

Our scheme can be used to make changes in the parameters of a Controller component at runtime. Since the environment and status of the system have various dynamic factors, the Controller component needs to adapt to them. The Algorithm Server provides the facility to upgrade an older Controller with a newer one which has a more advanced algorithm or a newer set of parameters at runtime, taking advantage of the runtime component management mechanism of Etherware.

4.4. Experiment: Model Adaptation Algorithm with Outlier Rejection

We illustrate the above features in the experiment described below. First, the experiment below demonstrates the presence of significant outlier errors in state estimation caused by jitter in packet delivery over the communication network in a vehicular networked control testbed. Then the suggested algorithm is applied to the networked system for a battery-powered car control, in the presence of outlier types of errors corrupting the velocity estimation, which are produced by the uncertainties of communication network. In the cyber-physical system implementation, the Filter

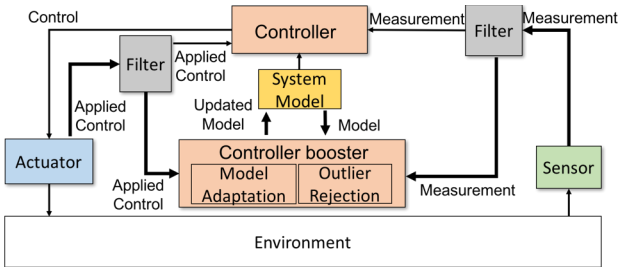


Figure 4.3: Control system configuration with model adaptation and outlier rejection.

mechanism of Etherware enables easy adoption of the outlier rejection algorithm and the model adaptation algorithm to improve control performance, both being accomplished without system reconfiguration.

We now describe the experiment performed in the vehicular networked control testbed. Figure 4.3 shows how outlier rejection and model adaptation are easily incorporated into the already pre-existing feedback control system by using the Filter mechanism of Etherware. The Sensor module obtains measurements about its environment and sends them to a Controller. The Filter Module between Controller and Sensor intercepts the measurements and sends them to an Outlier Rejection Module. It updates estimates after rejecting or modifying outliers in them, by making use of a system model. The Controller determines a control command based on the measurements and the system model, and sends it to an Actuator. The Filter between the Controller and the Actuator also serves another purpose. As it intercepts the control command to the Actuator, it also sends it to a Model Adaptation Module. The Model Adaptation Module then periodically updates the parameters of the system model using knowledge of the actual control commands and the estimates. Due to the removal of the outlier in the

estimates, as well as a more accurate system model, the controller provides more accurate control commands and thereby improves a system performance.

For the model adaptation algorithm with outlier rejection, an experimental scenario was designed as follows. A battery-powered car receives a rectangular trajectory from a High Level Controller. A Low Level Controller adopts a Model Predictive controller to control the car to follow the trajectory. As it consumes battery power, the system model of the controller and the real model of the relation between command indexes and car's velocities, diverge from each other. Hence, based on the observation data, the system model is periodically updated to adapt to the real battery power to improve control performance. While the observation data are transferred from a vision system through the network, random network delays cause wrong timestamps in the observation data, and outliers occur in the estimates of the velocity of the car. The proposed outlier rejection algorithm removes them by using the system model and a MMEM method in real time. The proposed model adaptation algorithm gathers up to 300 outlier-free estimates of the velocity of the car and then finds the best parameterized velocity model by a least squares method, to adapt to the real battery power. Recursively, a more precise system model is obtained by the outlier rejection algorithm to eliminate outliers better. Finally, the position and velocity errors of the car are highly improved.

We use a Model Predictive controller which minimizes a cost function:

$$\min_{i_{(k:k+N-1)}, j_{(k:k+N-1)}} J(x_k, x_k^d, i_{(k:k+N-1)}, j_{(k:k+N-1)}) \tag{12}$$

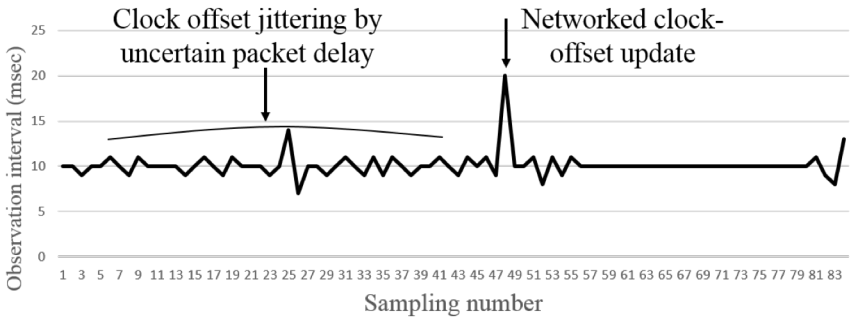
$$\text{s.t.} \quad x_{k+n+1} = f(x_{k+n}, w_{k+n}) + \hat{g}_{k+j}(x_{k+n}, v_{k+n})$$
$$v_{min} \le \hat{v}(i_{k+n}) \le v_{max}$$
$$w_{min} \le w(j_{k+n}) \le w_{max}$$

for all $n \in \{0,1,\ldots,N-1\}$,

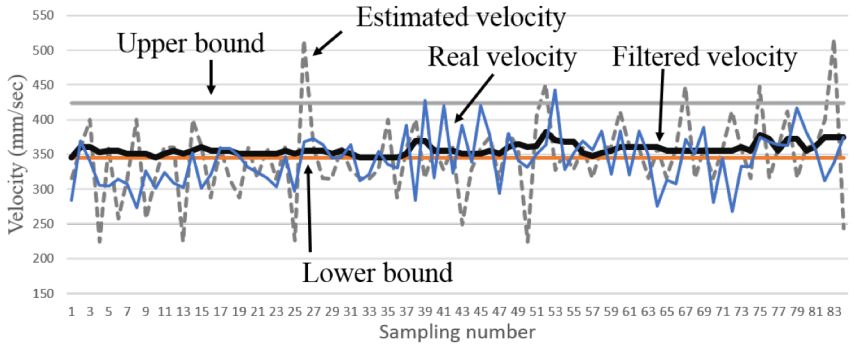
where at each time k, i_k and j_k are a velocity input and an angular velocity input, respectively, $J(\cdot)$ is a cost function, N is a control horizon, x_k and x_k^d are a state and a desired state from the trajectory, $v(\cdot)$ is an estimated system static nonlinearity model from observation data, and $w(\cdot)$ is assumed to be a time-invariant model. The controller obtains the position data from the vision system. The system model of the controller maps the discrete control levels to velocities. Based on the position data and the system model of the car, the controller determines control commands, and an actuator implements them to control the car to follow the desired trajectory.

The Outlier Rejection algorithm eliminates outliers in the velocity estimates by using a system model, and the Model Adaptation algorithm adapts the system model to the current battery power by using the filtered estimated velocity data. The velocity of the car is estimated based on the consecutive sequence of positions and timestamps of a car. Although the vision system of the testbed can determine fairly accurately the position of a car with high frequency, there still exist errors in the timestamps of the packets transferred through the network because of errors in time synchronization, packet delay or packet loss. These generate outliers in the estimation of the velocity of a car.

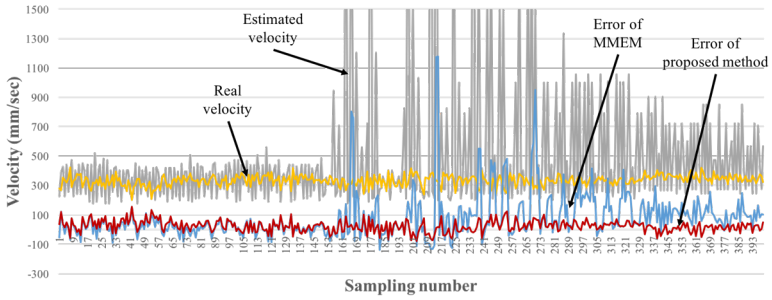
Figure 4.4 (a) shows how the communication network can generate outliers in the velocity estimates. When observation data are delivered through the network, their timestamp can be corrupted by clock offset jitter, by uncertain packet delay, and errors in runtime clock-offset synchronization. In Figure 4.4 (a), the observation interval is nominally supposed to be 10 msec, which corresponds to the vision system sending



(a) The variation of the observation interval based on the timestamp of packets



(b) Filtering and smoothing the estimated velocity with observation data containing outliers



(c) Comparison of the performances between the MMEM and the proposed method

Figure 4.4: Outlier rejection for velocity estimation of a car.

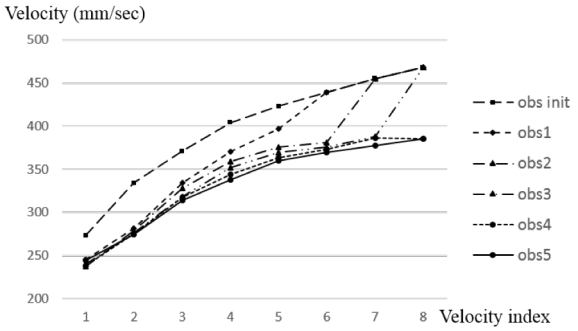
position data with timestamps to a controller every 10 msec. However, the interval between timestamps of position data which arrive at the controller suffers variation

because of the communication network. The corrupted observation data contaminates velocity estimates by generating outliers in them.

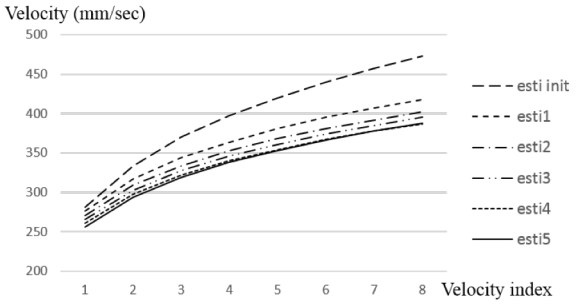
Figure 4.4 (b) demonstrates the results of outlier rejection. The dotted line is the graph of the raw data with outliers for the estimated velocity of a car moving according to a fixed velocity control command. The velocities are estimated with the position data and timestamps. The thick solid line in Figure 4.4 (b) represents the estimates filtered with a model of the car, which is truncated with upper bound and lower bound values from the model of a car, and smoothed by the MMEM filter. Clearly we have better estimates closer to real velocities after adopting the outlier rejection algorithm.

Figure 4.4 (c) shows the performance comparison between the MMEM and the proposed method. Though the real velocity stays at about 350 mm/sec, the estimated velocity has several outliers due to the congested network after 150 samples. The error graph of the MMEM shows that the MMEM can eliminate intermittent outliners well, but consecutive outliers still remain and cause huge errors. On the other hand, the error of the proposed method stays low even under the presence of severe outliers by filtering based on the system model. This demonstrates that the proposed method has better performance than MMEM.

In the Model Adaptation experiment, the initial velocity model is calibrated at high battery power but the real battery power is low. Thus, there is a discrepancy between the model and the real system. That is, for the same velocity control command, the real velocity of a car is slower than the velocity of the model. While the car follows a rectangular trajectory, the online model adaptation reduces the model mismatch. The



(a) Velocity estimates obtained from observations in each update iteration



(b) Adaptive model estimation with the velocity estimates filtered by outlier rejection

Figure 4.5: Model adaptation with velocity estimation.

updated velocity model gradually approaches the velocity model of the real battery power output. The decreased tracking error of the car demonstrates that the Model Adaptation algorithm improves the trajectory following performance of the controller. Figure 4.5 shows the online model adaptation for the velocity of a car. Forward movements correspond to velocity indices 1 to 8, with index 1 representing the minimum velocity and 8 the maximum. Figure 4.5 (a) shows that the average velocity estimates of each velocity index obtained from observations are updated at each step. The initial model has large maximum velocity, over 460 mm per sec, but the real battery power permits only a low maximum velocity under 390 mm per sec. At the first observation, the indices from 1 to 5

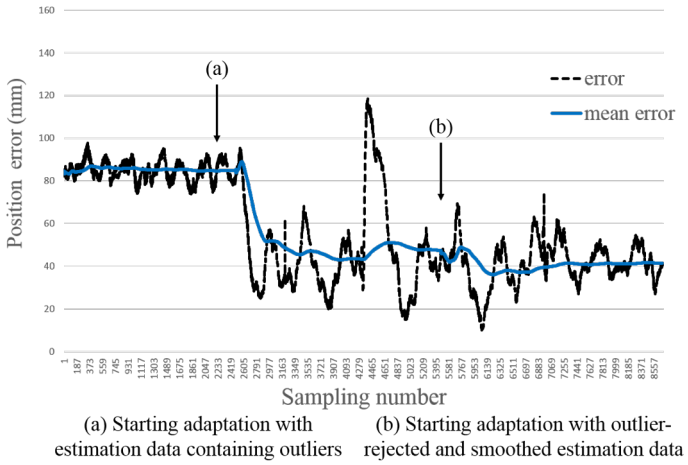
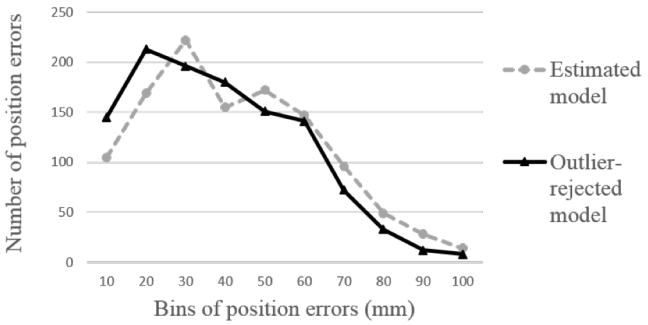


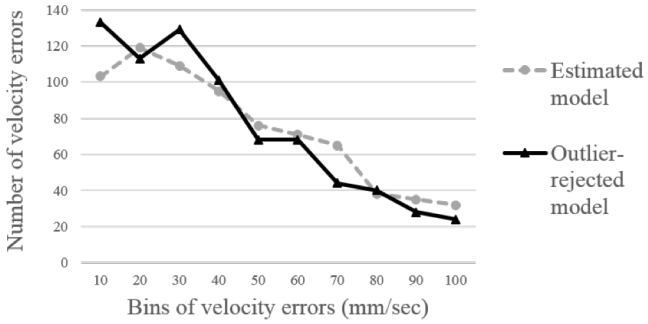
Figure 4.6: The reduced position errors of a car following a trajectory by runtime model adaptation algorithm with outlier rejection.

are updated by the estimations. At the fifth observation, all velocity indexes are updated. It shows that the real velocity of each index is lower than the initial velocity model that the controller has. In Figure 4.5 (b), each new velocity model is obtained based on each velocity-estimate curve from observation. At each iteration, the optimal parameters are determined from (11) to obtain the best estimated velocity model. The updated velocity model approaches the real velocity model corresponding to the current low battery power.

Figure 4.6 shows the position errors of a car following a trajectory. The position error is the deviation of car's real position from the desired position at the desired arrival time of the trajectory. Due to the difference between the velocity model used by its controller and the real battery status, the car's controller cannot reduce its position error below a certain value before applying the adaptation algorithm. At time (a), the adaption algorithm without outlier rejection is applied to the controller at runtime by the Filter mechanism of Etherware. The position errors are reduced, but they have large variation



(a) Histograms of position errors as desired the probability of error is reduced for large position errors.



(b) Histogram of velocity errors as desired the probability of error is reduced for large velocity errors.

Figure 4.7: The comparison between the histograms of position errors and velocity errors for an estimated adaptive model and an outlier-rejected adaptive model, showing the lower probability of error for large position and velocity errors.

because the adaptive model can hardly converge due to the outliers in the estimates. After the adaptation starts using the outlier-rejected and smoothed estimates, the controller achieves a more stable error boundary and reduced mean error.

Figure 4.7 compares the control performance of a normal controller employing an adaptive model with outlier-containing estimation, versus a controller employing outlier rejection and model adaptation. It shows the histograms of car's position errors and velocity errors for each controller. The histogram in Figure 4.7 (a) shows how many

Table 4.1: The percentages of frequencies in the histograms

(%)	Position error		Velocity error	
	Below 50 mm	***Over 50 mm***	***Below 50 mm/sec***	***Over 50 mm/sec***
Outlier-rejected estimation	76.89	23.11	72.73	27.27
contaminated estimation	71.13	28.87	67.56	32.44

position errors break out in each range of the bins. The bins of position errors are from 10 mm to 100 mm. The solid line with triangles is the frequency graph for an outlier-rejected model and the dotted line with circles for an estimated model. The controller using an adaptive model with outlier-rejected estimation has more number of position errors than one using with estimation contaminated by outliers in the low range of the position error and less in the high. It means that the outlier-rejected model adaptation can guarantee lower trajectory error than the adaptive model with outlier-containing estimation. We can also see the better performance of the suggested algorithm in the view of the velocity error of a car in Figure 4.7 (b). The velocity error is the deviation of the current velocity from the desired velocity determined by the current velocity model and the last command. The number of velocity errors for an outlier-rejected model is greater than for an estimated model for velocity errors below 50 mm/sec but is lesser over 50 mm/sec. Table 4.1 shows the percentages of frequencies in the histograms. The values are calculated from the ratios of the frequency of low errors to one of high errors for each adaptive model. The percentage of the position errors below 50 mm with outlier-rejected estimation is 76.89% of all the position errors which is 5% greater than with outlier-contaminated estimation,

as the same in velocity error. The experimental results demonstrate that the suggested model adaptation with outlier rejection improves the tracking performance of a car controller by reducing both position errors and velocity errors, effectively.

5. ADVANCED APPLICATIONS OF CYBER-PHYSICAL SYSTEMS*

In this chapter we consider two advanced applications of cyber-physical systems. One concerns the problem of security of autonomous transportation systems, a topic of great current interest as previously closed safety-critical systems are opened to the Internet. The second application considers cyber-manufacturing based on origami-style folding and cutting of sheet material. It can potentially result in a faster manufacturing system than additive manufacturing that requires a large amount of material volume needing to be printed layer by layer.

5.1. Implementation of Dynamic Watermarking for Cybersecurity of Advanced Transportation Systems

Recently, there has been increasing interest in developing fully autonomous cars and also in systems of autonomous vehicles. However, accompanying these capabilities, are safety issues for automated vehicles that are also emerging, as many incidents have shown that automated cars can be attacked by malicious agents. It is therefore of great interest to determine methods by which advanced transportation systems can be made secure to attacks from hackers in [29]-[33]. More broadly the problem of security of cyber-physical systems is acquiring great importance as critical infrastructure is opened to the Internet or wireless networks.

We will consider the method of "dynamic watermarking." This is an active defense technique for CPSs to detect malicious nodes, based only on local information available

*Reprinted with permission from "Theory and implementation of dynamic watermarking for cybersecurity of advanced transportation systems" by Woo-Hyun Ko and P. R. Kumar, 2016. *IEEE CNS*, pp. 416-420, Copyright 2016 by IEEE

to honest nodes in distributed systems [34]-[37][42][43]. The technique consists of honest nodes adding a secret random signal as a watermark to the nominal actuation signal. The watermark is not made available to other nodes, and is kept small enough not to degrade the performance of the system. Since the actuation signal enters a feedback loop, the actuators can detect contaminated data in their inputs from other nodes if the secret watermark in not present in their returned signals.

The dynamic watermarking technique has been introduced to effectively detect malicious sensors in a linear system in previous work [34]-[38]. Two tests can be employed for dynamic watermarking to detect data manipulated by malicious sensors. In this section, we extend the technique of dynamic watermarking to nonlinear systems modeling vehicles, so that it can secure an intelligent transportation system by preventing collisions caused by a malicious sensor node.

5.1.1. Dynamic Watermarking

Dynamic watermarking can be used by actuator nodes to detect malicious sensor nodes. An actuator node generates an independent identically distributed random process with a normal distribution which has zero mean and variance small enough not to degrade system performance, for use as a watermark, and adds it to the control command at every actuation sampling time. The watermark within the control command affects the plant's output. As a result, the observation data of the sensor node contain a suitably transformed version of the random signal within it. The actuator node investigates the statistics of the signal returned by the sensor node, by performing two statistical tests to verify the existence of the watermark within the returned signals. If it detects an absence of the

watermark, or a version different from the transformed version that is expected to be returned due to the model of the plant, then it regards the sensor node as a malicious node.

We begin by considering a linear system to describe how the two tests work to detect the absence of a watermark. Let y[t] denote a system output, z^t, the past of the observation information, (A, B) a system model, u_t^g the control input with a control law g, and $w[t]$ a system noise which is white Gaussian noise. The watermark of the i^{th} actuator node is $e_i[t]$, with $e[t] \sim \mathcal{N}(0, \sigma_e^2 I)$. The overall linear system evolves as

$$y[t+1] = Ay[t] + Bu_t^g(z^t) + Be[t] + w[t+1]. \tag{5.1}$$

Therefore, the real output of the system satisfies the following probabilistic relations

$$\{y[t+1] - Ay[t] - Bu_t^g(z^t)\} \sim \mathcal{N}(0, BB^T\sigma_e^2 + \sigma_w^2 I), \tag{5.2}$$

and

$$E\left[e_i[t]\left(y[t+1] - Ay[t] - Bu_t^g(z^t)\right)\right] = B_{.i}\sigma_e^2, \tag{5.3}$$

where $B_{.i}$ denotes the i^{th} column of B.

Based on the above observations, two tests are defined for an honest actuator node to check if the measurement contains a watermark or not:

1) Test 1:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{k=0}^{T-1} \left(z[k+1] - Az[k] - Bg_k(z^k)\right)\left(z[k+1] - Az[k] - Bg_k(z^k)\right)^T$$

$$= \sigma_e^2 BB^T + \sigma_w^2 I_n$$

2) Test 2:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{k=0}^{T-1} e_i[k]\left(z[k+1] - Az[k] - Bg_k(z^k)\right) = B_{.i}\sigma_e^2$$

The measurement $z[k+1]$ from the sensor node must pass both Test 1 and Test 2. If it fails one of the tests or both, that indicates the absence of the watermark in the measurement from the sensor node due to malicious attacks.

5.1.2. Protecting Autonomous Transportation Systems from Malicious Attacks on a Sensor with Dynamic Watermarking

Autonomous transportation systems consist of several different mobile nodes each perceiving its environment and controlling its own vehicle to track its desired routes safely. Communication networks are used by the nodes to exchange information with one another. However, some of the nodes may be vulnerable to attacks from external malicious nodes through the networks. Such a malicious attack can cause vehicles to collide, which constitutes a serious threat to intelligent transportation systems. We experimentally demonstrate how dynamic watermarking can detect malicious sensors and prevent vehicles collisions.

Let (x_i, y_i, θ_i) denote the state of the i^{th} vehicle, (v_i, w_i) the velocity and angular velocity of a control input, $(\omega_{iv}, \omega_{iw})$ the velocity noise and angular velocity noise in a control noise. Every h time units, the system state is updated as

$$x_i[t+1] = x_i[t] + h \cdot cos(\theta_i[t])(v_i[t] + \omega_{iv}[t]), \tag{5.4}$$

$$y_i[t+1] = y_i[t] + h \cdot sin(\theta_i[t])(v_i[t] + \omega_{iv}[t]), \tag{5.5}$$

$$\theta_i[t+1] = \theta_i[t] + h \cdot (w_i[t] + \omega_{iw}[t]). \tag{5.6}$$

The i^{th} actuator node moves the i^{th} vehicle by executing the control input $(v_i[t], w_i[t])$, and the i^{th} sensor node measures the state of the i^{th} vehicle. We assume that the second sensor node is under attack and it starts manipulating the x-position data

of the second vehicle commencing at the time t_A. The attack consists of adding a bias τ to the new measurement of the x-position as in the following.

$$z_{2x}[t] = x_2[t] + \tau, \tag{5.7}$$

where $t > t_A$. The malicious sensor node then reports the measurements $\{z_{2x}\}$ generated as

$$z_{2x}[t+1] = z_{2x}[t] + h \cdot \cos(\theta_2[t])\, u_{2v}^{g}(z_1^t, z_2^t) + n[t], \tag{5.8}$$

where $n[t] \sim \mathcal{N}(0, \sigma_x^2)$, and $u_{2v}^{g}(z_1^t, z_2^t)$ denotes the control policy-specified input for the input $v_i[t]$.

On the other hand, in order to detect a malicious attack on a sensor node, the system with dynamic watermarking evolves as

$$x_2[t+1] = x_2[t] + h \cdot \cos(\theta_2[t]) \left(u_{2v}^{g}(z_1^t, z_2^t) + e_{2v}[t] + \omega_{2v}[t]\right), \tag{5.9}$$

$$y_2[t+1] = y_2[t] + h \cdot \sin(\theta_2[t]) \left(u_{2v}^{g}(z_1^t, z_2^t) + e_{2v}[t] + \omega_{2v}[t]\right), \tag{5.10}$$

$$\theta_2[t+1] = \theta_2[t] + h \cdot \left(u_{2w}^{g}(z_1^t, z_2^t) + e_{2w}[t] + \omega_{2w}[t]\right), \tag{5.11}$$

where $e_{2v}[t] \sim \mathcal{N}(0, \sigma_{e_v}^2)$ and $e_{2w}[t] \sim \mathcal{N}(0, \sigma_{e_w}^2)$ are the watermark in the velocity control and the angular velocity, respectively.

Due to the attack by the malicious sensor node, the manipulated sensing information $z_2[t]$ is described by

$$z_2[t] = \left[z_{2x}[t], z_{2y}[t], z_{2\theta}[t]\right]^T = \left[z_{2x}[t], y_2[t], \theta_2[t]\right]^T. \tag{5.12}$$

This is investigated by the two below tests at every sampling time:

1) Test 1:

$$\lim_{t\to\infty}\frac{1}{t}\sum_{k=0}^{t-1}(z_{2x}[k+1]-z_{2x}[k]-h\cdot\cos(z_{2\theta}[k])\,u_{v_2}^{g}(z_1^t,z_2^t)-h\cdot$$

$$\cos\,(z_{2\theta}[k])e_{2v}[k])^2\;=\tilde{\sigma}_x^2 \tag{5.13}$$

2) Test 2:

$$\lim_{t\to\infty}\frac{1}{t}\sum_{k=0}^{t-1}(z_{2x}[k+1]-z_{2x}[k]-h\cdot\cos(z_{2\theta}[k])\,u_{v_2}^{g}(z_1^t,z_2^t))^2\;=\sigma_c^2 \tag{5.14}$$

where

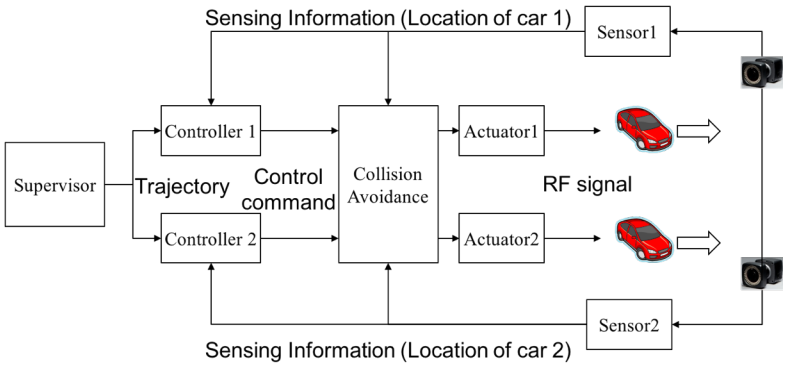
$$\tilde{\sigma}_x^2 \coloneqq \lim_{t\to\infty}\frac{1}{t}\sum_{k=0}^{t-1}(h\cdot\cos(\theta_2[k])\,\omega_{2v}[k])^2,$$

and

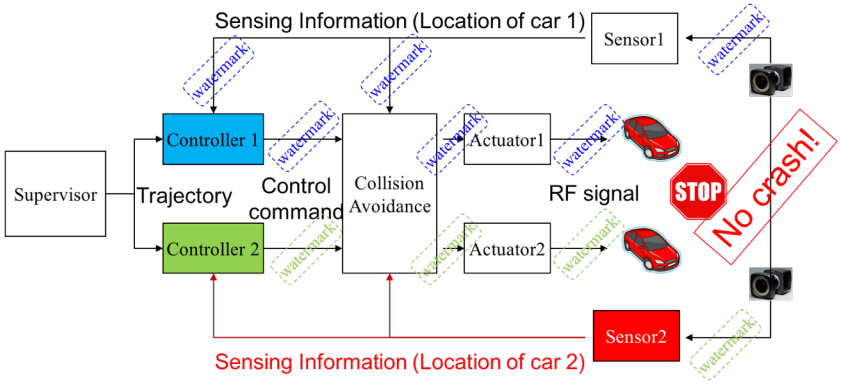
$$\sigma_c^2 \coloneqq \lim_{t\to\infty}\frac{1}{t}\sum_{k=0}^{t-1}(h\cdot\cos(\theta_2[k])\,e_{2v}[k]+h\cdot\cos(\theta_2[k])\,\omega_{2v}[k])^2,$$

It is assumed that an attack on a sensor node commences sometime after the system has started. The RHSs of the two tests are verified by the statistics which have been obtained before the attack. If the newly updated sensing information fails one or both tests, the actuator node judges it as being contaminated, and concludes that the sensor node is malicious.

Figure 5.1 (a) shows the flow of control and information among the nodes in the control of two cars. A supervisor node generates trajectories and sends them to the controller nodes. Controller nodes determine the control commands and a collision avoidance node detects an impending collision by investigating the control commands. Actuator nodes trigger their respective cars to move and sensor nodes give back sensing information as feedback.



(a) System configuration of an autonomous transportation system with collision avoidance



(b) Dynamic watermarking prevents collisions between vehicles by detecting a malicious sensor

Figure 5.1: System configuration and watermarking.

If the sensor node 2 is malicious and disseminates wrong sensing information to the controller node 2 and the collision avoidance node, then car 2 will deviate from its trajectory. Consequently, even the collision avoidance node will not be able to predict or prevent any collision between the cars. As a result, the malicious sensor node 2 can cause car 2 to crash into the other car. This is an example of a potentially serious threat to intelligent transportation systems.

In order to secure the system, we implement dynamic watermarking in the controller nodes. Figure 5.1 (b) describes how the dynamic watermark secures the safety of the cars against the malicious sensor by preventing collisions. At every control sampling time, each controller generates a random number with a normal distribution which has zero mean and a specified small value of variance as a watermark, and injects it as an addition to the control command. Each watermark flows around the closed loop. If the malicious sensor node 2 manipulates the sensing information of the car 2, the watermark disappears from the sensing information or is distorted in a detectable way. As soon as the controller node 2 detects the absence/distortion of the watermark by the change of the variance of the errors, it stops all cars to prevent them from having a collision.

Figure 5.2 shows the runtime statistics obtained from the experiment. At time 60 seconds, the sensor node is attacked and the watermark disappears from the manipulated sensing information, so the variances of the error tests change. The variance of error test 2 remains the same, and passes. However, the variance of error test 1 increases beyond the

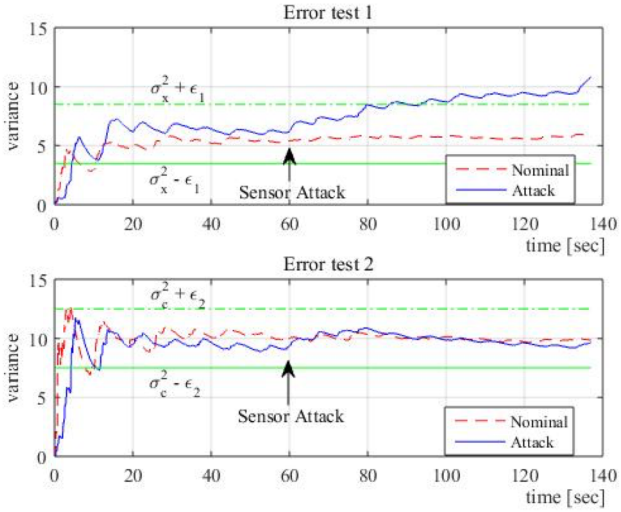


Figure 5.2: Test statistics of error tests 1 and 2 as a function of time.

allowed threshold, so it fails. Therefore, the actuator node detects the attack on the sensor node.

5.2. A Multi-Component Automated Laser Origami System for Cyber Manufacturing

We now investigate another advanced application of cyber-physical systems – cyber-manufacturing. We consider a system employing an origami-like strategy of folding and cutting material to manufacture 3D objects. This is potentially a quicker alterative to additive manufacturing.

This section describes the implementation of such a novel custom manufacturing machine system. By contrast with high volume manufacturing, a custom manufacturing system vitally requires real-time planning, adaptation and control. Moreover, low volume manufacturing also needs fault monitoring and control, as well as process planning. Hence, control schemes must be developed to prevent process failures by employing state prediction, in addition to securing and guaranteeing the safety of the system.

While there has been much research about process planning in the forming operation including the study of origami folding robots, the proposed approach features the integration of laser heating and incremental forming processes, as well as the feedback-based control.

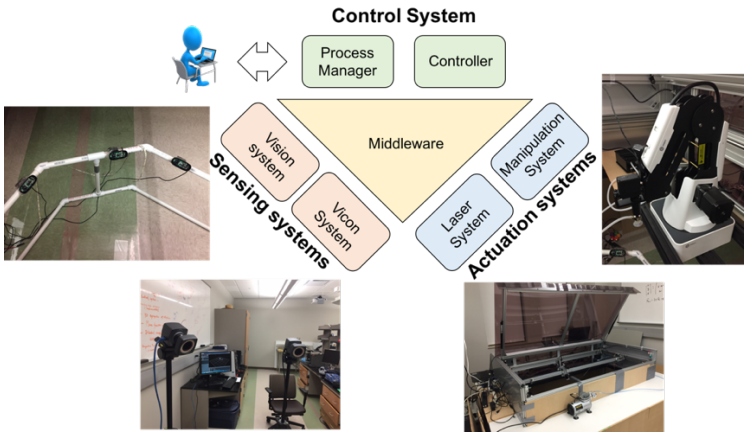


Figure 5.3: The overview of an automated laser origami system based on middleware.

We propose an automated laser origami system for cyber manufacturing. Figure 5.3 shows the overview of an automated laser origami system based on middleware. A sensing system with cameras monitors the status of the manufacturing process and a control system controls the feed rate of a laser head, the intensity of a laser beam and the position of a robotic arm based on the feedback information from the sensing system to operate cutting and bending processes, automatically. Middleware enables laser, sensing and control systems, which are connected over a network, to communicate with one another by sending and receiving messages. It can also support component-based development which offers simple quick deployment, easy extension, and maintenance.

We validate the proposed system by experimenting with laser cutting and bending in a real laser origami system.

5.2.1. Related Work

Balkcom et al. [53] extended some well-known necessary conditions for origami patterns to be flat-foldable, such as Kawasaki's theorem, Maekawa's theorem, and Meguro's theorem, for automatic planning and folding manipulation, and presented the world's first origami folding robot through a video. However, all processing was open-loop so that the number of foldings was limited due to the accumulated errors in the frictional insertion of the paper into the slot and in the positioning and grasping of the paper. Namiki and Yokosawa [54] achieved a feedback-based robotic origami folding in a real system. They extracted dynamic motion primitives from a valley folding process and experimentally demonstrated valley folds twice in a row in real time by using 3D visual and force-torque information and a physical simulation model. On the other hand, there was still a limit to the kind of a material that could be only a flexible sheet, like a paper. Mueller et al. [51][52] suggested a prototyping system which could manufacture a 3D object by using a regular laser cutter. A laser was defocused to heat a desired region on the surface of a workpiece until it was bent or stretched to produce a 3D object. They proposed three design elements by using Laser-Origami and presented various 3D creations such as a card holder, a suspender, and a pen holder, which were manufactured by Laser-Origami without manual assembly. However, the open-loop process required an operator to supervise the bending process to achieve the desired angles.

5.2.2. Middleware-based System Integration

Current machine tool monitoring technologies are mostly limited to using static or well-accessed wireless sensors. Recent advances in mobile communications and computer vision algorithms offer a radically new approach for real-time process control and quality assurance. We have built a multi-component manufacturing system centered around a laser processing machine. The network architecture for the manufacturing system offers considerable advantages for custom manufacturing systems. For the system integration, Etherware was used to implement the networking among the individual system components [1][4][26][28][44]-[46]. The middleware enables control over a communication network and computations can be carried out elsewhere in the cloud with seamless migration.

Figure 5.4 shows the feedback-based laser origami system. All systems are connected over Ethernet and communicate with each other by Etherware which supports message-based communication. The kernel of Etherware, which is run in each computer,

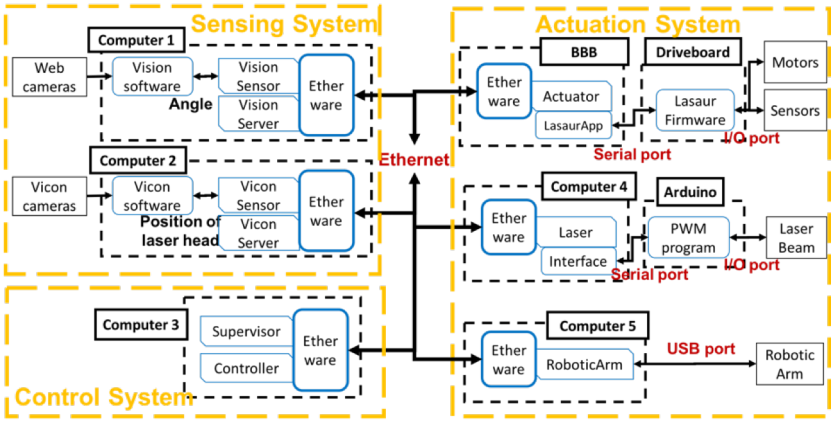


Figure 5.4: The system configuration of the automatic laser origami control system based on middleware.

executes components, and the profile registry service of Etherware provides a semantic naming service with the profile table mapping IP addresses of computers and IDs of components which are assigned by the kernel to the profile names of components. For example, a Vision Sensor component can send a message containing the bending angle to a Controller component by using only its name 'Controller', instead of the IP address of the computer which the component is running on. Networking among the system components enables the scalability of the system, Etherware supports component-based application developments and facilitates not only the development of the individual system components, but also system integration, management, and maintenance.

The proposed automatic manufacturing system consists of three subsystems: a sensing system, a control system, and an actuation system. The sensing system observes the status of the process and provides observed information to a control system. For the sensing system, we have used two individual vision systems consisting of a multi-camera vision system which finds the bending angle of the workpiece during the bending procedure, and a Vicon system which finds the position and velocity of the laser head. Vision and Vicon sensor components obtain observation data from image processing programs, and Server components deliver them to a Controller component.

The control system has a supervisor as a high-level process manager and a bending controller. The supervisor receives the list of tasks from an operator and sends each task to a bending controller or an actuator system. With the exception of bending tasks all tasks such as cutting tasks or motion tasks of a robotic arm are executed by the actuator system directly because they do not need any feedback. When the current task is a bending task,

it is executed by a bending controller. Once the bending controller receives the desired angle, it determines Gcode-type commands describing the intensity of the laser beam, and the feed rate of the laser head, based on the feedback from the sensing system, and sends them to the actuator system.

The actuator system plays the role of moving the laser head, regulating the intensity of the laser beam and manipulating the robotic arm. It executes the sequence of gcodes received from the high-level process manager and the bending controller to manufacture a 3D product from a sheet material by cut-bend-fold operations.

5.2.3. Feedback Control Scheme for Laser Bending

A feedback-based laser bending control system has shortened manufacturing lead time. Since manual operation for laser bending uses constant intensity of laser beam and feed rate of laser head throughout the whole bending process, it demands much trial and error to find the best heating policy manually. Especially, in the case of the changes of some parameters such as kind of work sheet or length of a bending line, a new heating policy will need to be found by trial and error. Feedback-based control system automatically deals with those issues. Based on the latest observation data, a control policy determines control inputs such as the feed rate of laser head and intensity of laser beam every control period, so that it not only accomplishes fast bending, but also prevents overheating which can degrade the quality of the bending process.

For laser bending, the feedback system controls the feed rate of the laser head and the intensity of the laser beam. A PID controller determines control inputs based on the

desired angle from a supervisor component and the previous bending angle from the sensing system.

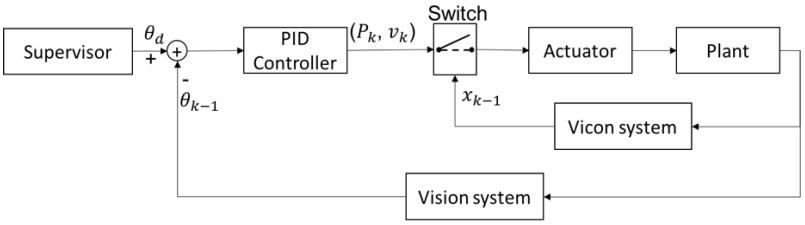


Figure 5.5: The control scheme for laser bending.

Figure 5.5 shows the control scheme with a PID controller for laser bending, where θ_d is the desired bending angle, θ_{k-1} is an observed bending angle at time $k-1$, and P_k and v_k are the intensity of a laser beam and the feed rate of a laser head, respectively, at time k. Let $e_{k-1} = \theta_d - \theta_{k-1}$ denote the error of the bending angle at time $k-1$. The PID controller for calculating the desired derivative of the intensity of the laser beam ΔP_k is designed as

$$\Delta P_k = k_{P_P} \cdot e_{k-1} + k_{I_P} \cdot \sum_{i=0}^{k-1} e_i \cdot dt_i \quad + k_{D_P} \cdot \left(\frac{e_{k-1} - e_{k-2}}{dt_{k-1}}\right) - P_{k-1}, \quad (5.15)$$

where k_{P_P}, k_{I_P} and k_{D_P} are the proportional, integral and derivative gains, respectively. The intensity of the laser beam at time k is determined as

$$P_k = \begin{cases} P_{k-1} + P_{step}, & if\ \Delta P_k > P_{step} \\ P_{k-1} - P_{step}, & if\ \Delta P_k < 0 \\ P_{k-1}, & otherwise \end{cases}, \quad (5.16)$$

where P_{step} is the step size for increasing or decreasing the intensity of the laser power. In order to design a controller for the feed rate of the laser head, let $\dot{e}_{k-1} = (e_{k-1} - e_{k-2})/dt_{k-1}$ denote the derivative of the error of the bending angle. The derivative portion of the controller for calculating a desired derivative feed rate is designed as

$$\Delta v_k = k_{d_v} \cdot (\dot{e}_{k-1}) + v_{base} - v_{k-1}, \tag{5.17}$$

where k_{d_v} is a derivative gain and v_{base} is the minimum feed rate. The feed rate of the laser head at time k is determined as

$$\mathrm{v}_k = \begin{cases} v_{k-1} + v_{step}, & if\ \Delta v_k > v_{step} \\ v_{k-1} - v_{step}, & if\ \Delta v_k < 0 \\ v_{k-1}, & otherwise \end{cases}, \tag{5.18}$$

where v_{step} is the step size for increasing or decreasing the feed rate of the laser head. The control inputs P_k and v_k are delivered to an actuator component every control interval. The control interval is determined as

$$dt_k = \frac{2 \cdot L_{bending}}{v_{k-1}}, \tag{5.19}$$

where v_{k-1} is the previous feed rate of the laser head and $L_{bending}$ is the length of a bending line. This means that the control interval is varying during bending control process. When the previous feed rate of the laser head is slow, the current control interval is long. Conversely, a fast previous feed rate makes the control interval short. A flexible interval enables the control decision timing to adapt to the status of the bending. For example, when the angular velocity of bending is zero, the bending line is heated up radically with slow feed rates of the laser head. When the unfixed segment of the part starts falling down, the bending angle is controlled more frequently with fast feed rate of the laser head. This allows achievement of more precise bending, even mid-air stopping at 60 degrees.

The switch obtains the position data of the laser head from a Vicon system to check if the back and forth movement of the laser head is completed. When the laser head returns back to the starting position, the next control input is delivered to the actuator. According

to the actuator's output signals, the laser system bends the material by the movement of the laser head and the heat from the laser beam. When the bending angle reaches the desired angle, the bending process is terminated.

5.2.4. Experiment

This experiment validates the performance of the proposed feedback-based control system for cyber-manufacturing. Existing laser bending processes require an operator to manually monitor the status of the bending angle and repeated by heat the bending line by sending commands. Hence, the bending process is separated from a cutting process. This has several inherent issues such as the possibility of overheating caused by human errors, the dependency of the process performance on the skill level of the operator, and repeated trial and error experimentation to obtain better performance. On the other hand, the proposed feedback-based control system eliminates the need of an operator to carry out the repetitive tasks. Moreover, it results in uniform and proper heating, and minimizes the deformation of the bending line by controlling the heat output of the laser and the velocity of the laser head in real time. It results in high performance and quality of the bending process. The performance of the proposed system is compared with one of a manually bending system.

The experiment of evaluation validates the performance of the proposed feedback-based control system for cyber-manufacturing. Existing laser bending processes require an operator to manually repeat the sending commands to heat the bending line or stop the bending process while monitoring the status of the bending angle. This has several inherent issues such as the separation of the bending process from cutting processes, the

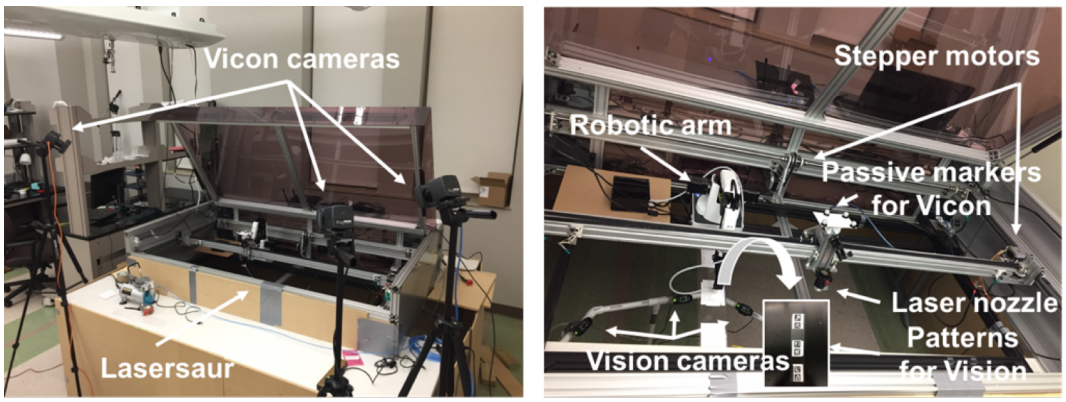


Figure 5.6: The overview of the experimental testbed.

possibility of overheating caused by human errors, the dependency of the process performance on the skill level of the operator, and repeated trial and error experimentation to obtain better performance. On the other hand, the proposed feedback-based control system eliminates the need for an operator to carry out the repetitive tasks. Moreover, it results in uniform and proper heating, and minimizes the deformation of the bending line by controlling the heat output of the laser and the velocity of the laser head in real time. It results in high performance and quality of the bending process. The performance of the proposed system is compared with that of a manually bending system.

Figure 5.6 shows a feedback-based laser origami control system which controls a laser system to manufacture a 3D product by cut-bend-fold operations with a sheet material. The cyber-manufacturing system consists of a vision system estimating the state of the workpiece, a Vicon system observing the position and velocity of a laser head, a control system determining Gcode-type control inputs to a laser system based on the estimated data and user's job requests, a robotic arm manipulating the workpiece, and a laser system processing the workpiece with an incremental forming tool and a laser source.

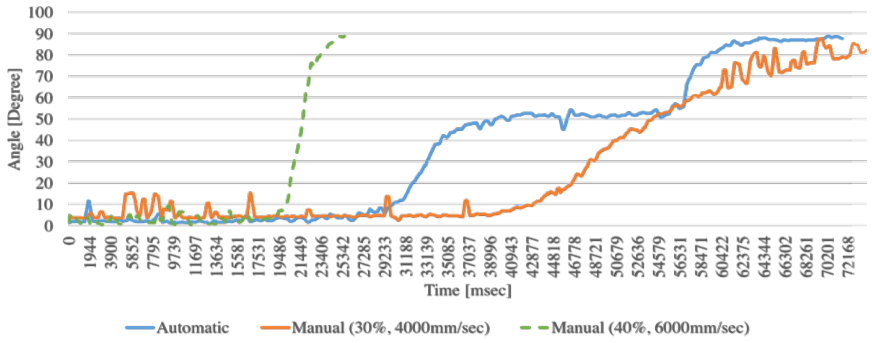
The laser system is Lasersaur [57]. Its maximum laser power is 120W. It has two stepper motors to move a laser head with the maximum speed 6000 mm/min on the work area 1220 x 610 mm. The resolution of the movement of a laser head is less than 0.1 mm.

The robotic arm is Dobot magician [58]. It has 4 degrees of freedom. It can move the end effector along the x, y and z axes in the work area which have dimensions, 340 x 300 x 400 mm, and rotate up to 270 degrees. The payload is 500 grams.

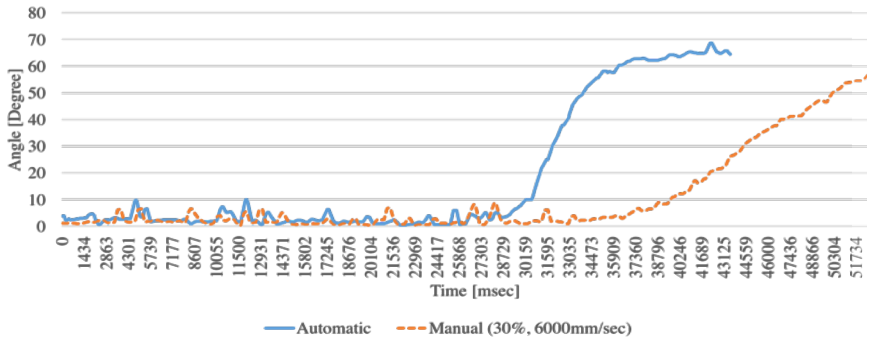
A multi-camera vision system uses three web cameras whose resolution is 640 by 480 pixels. The cameras detect two dimensional barcode pattern patches which are attached on the surface of the workpiece. The vision program calculates the angle between the two planes, defined by the patterns, every 150 milliseconds. Moreover, we used three Vicon cameras, which are LED cameras, to detect the shape of a group of small balls on the top of the laser head every 100 milliseconds. The information is used for the control system to determine when the control commands should be executed because of the varying control period during the bending procedure.

We experimentally demonstrate that the proposed system noticeably improves the performance of the process and the quality of the product. Figure 5.7 and Figure 5.8 compare the performances of those manually bent versus those automatically bent, with respect to completion time and quality of bending.

Figure 5.7 describes the graphs of the bending angles with time. The dimension of work piece used in the experiment was 355 x 70 x 1 mm. The length of the bending line



(a) 90 degree bending

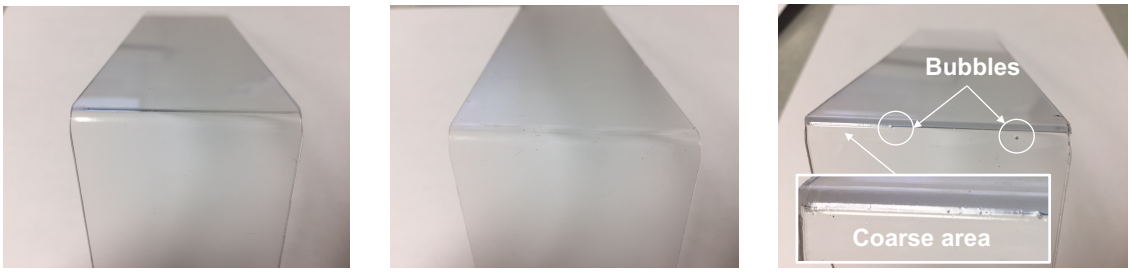


(b) 60 degree bending

Figure 5.7: The comparison of the performances of automatic bending and manual bending from the viewpoint of completion time.

was 100 mm, which was longer than the width of the work piece to avoid overheating at both end sides by laser. The focal distance from the laser nozzle to the work piece for bending was 150 mm. For the manual bending, we used two pairs of static intensities of the laser beam and feed rates of the laser head. They are 30% with 4000 mm/sec, and 40% with 6000 mm/sec.

In Figure 5.7 (a), when we used the pair of 40% and 6000 mm/sec, the bending angle, shown by the big dotted line, dramatically reached the desired degree with 8 back



(a) Automatically bent (b) Manually bent (30%, 4000 mm/sec) (c) Manually bent (40%, 6000 mm/sec)

Figure 5.8: The comparison of the performances of automatic bending and manual bending from the viewpoint of quality of bending.

and forth laser heatings. However, Figure 5.8 (c) shows that radical heating by the strong laser beam degraded the quality of the bent surface by causing some bubbles and coarse area. Likewise, although Figure 5.8 (b) shows that the manual bending with 30% intensity and 4000 mm/sec feed rate achieved high quality of bending surface, its completion time for bending was too long, which is indicated by the small dotted line shown in Figure 5.7 (a), with 20 back and forth laser heatings. Otherwise, the automatic bending, shown by the solid line in Figure 5.7 (a), accomplished an earlier completion time than manual bending with 30% and 4000 mm/sec as well as better quality of bending than manual bending with 40% and 6000 mm/sec in Figure 5.8 (a).

The high performance of the automatic bending is also shown in Figure 5.7 (b). Without a rotation machine, in order to reach a mid-air angle which is not 90 degrees, an operator needs to keep checking current bending angle and stop the back and forth heating as quickly as possible when the current bending angle is near the desired angle. If the feed rate is set too fast to minimize the response time, it also causes a delay of completion time.

Figure 5.7 (b) shows that the completion time of the automatic bending is much faster than one of manual bending with 30% and 6000 mm/sec, to reach 60 degrees.

We also demonstrate that the proposed automatic laser manufacturing system performs cut-bend operations effectively. An operator mounts a work sheet on the robotic arm and executes the gcode file which includes cutting and bending tasks. The system automatically manages and controls the whole process to fabricate a 3D-shaped creature. We used an acrylic sheet whose size is 355 x 279 x 1 mm. Figure 25 shows the entire procedure of cutting and bending.

An automatic laser fabricating system makes a simple 3D creation by performing 3 cutting and 2 bending processes. In Figure 5.9, 'p' is the holding position of the robotic arm, 'a', 'b' and 'c' are lines for cutting, and 'd' and 'e' for bending. At first, a laser system cuts the three cutting lines and then, bends along the two bending lines. For cutting, the robotic arm lifts the acrylic sheet up to the focal length of cutting. After cutting the three lines, the outer piece of the sheet is removed. As the bending process requires a long focal length of the laser to melt the bending line, the robotic arm lowers the sheet from the laser

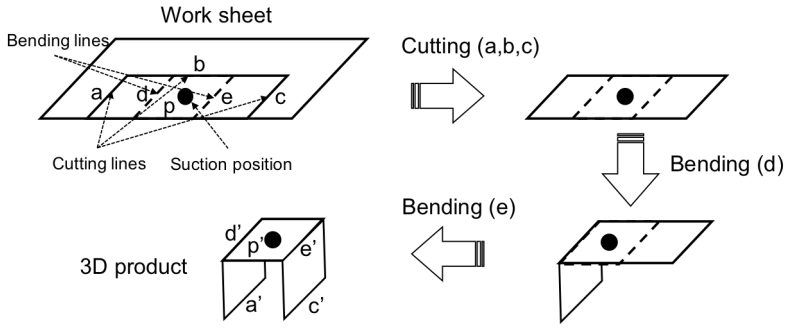


Figure 5.9: The procedure for 3 cutting and 2 bending processes.

nozzle. The desired bending angle is 90 degrees for the both bending processes. During the bending process, a control system controls the bending angle to reach the desired value by sending control inputs such as the intensity of the laser beam and the feed rate of the laser head in real time. When the temperature of the heated region is high enough to melt the workpiece, the unattached portion of the workpiece is pulled downward by gravity, and multiple cameras measure the angle of the falling part and send the information to the control system as feedback. After the first bending process, the robotic arm rotates the workpiece to place the other side in the bending position. Once all cutting and bending processes are completed, we obtain the 3D product as in the Figure 5.9**.** The automatic two-sided bending experiment can be seen in [59].

6. PROBABILITY-BASED COLLISION DETECTION AND RESOLUTION FOR UNMANNED AERIAL SYSTEM TRAFFIC MANAGEMENT

Many recent studies focusing on drones and advanced technologies have identified several potentially important applications including package delivery, videography, news gathering, infrastructure surveillance, agriculture and firefighting. The bottom most portion of the airspace from 200 feet to 500 feet above ground level is being contemplated operation of such drones. As their numbers multiply, there is a critical need to secure safety, manage air traffic, and ensure timely travel of drones [47], all to be done in an automated manner.

The entire autonomous system is envisaged to consist of a collection of enterprises, each possessing some number of drones. Individual companies plan the nominal trajectories for the operations of their drones. However, private companies do not need to share information about the planned trajectories with other companies' drones. Real-time

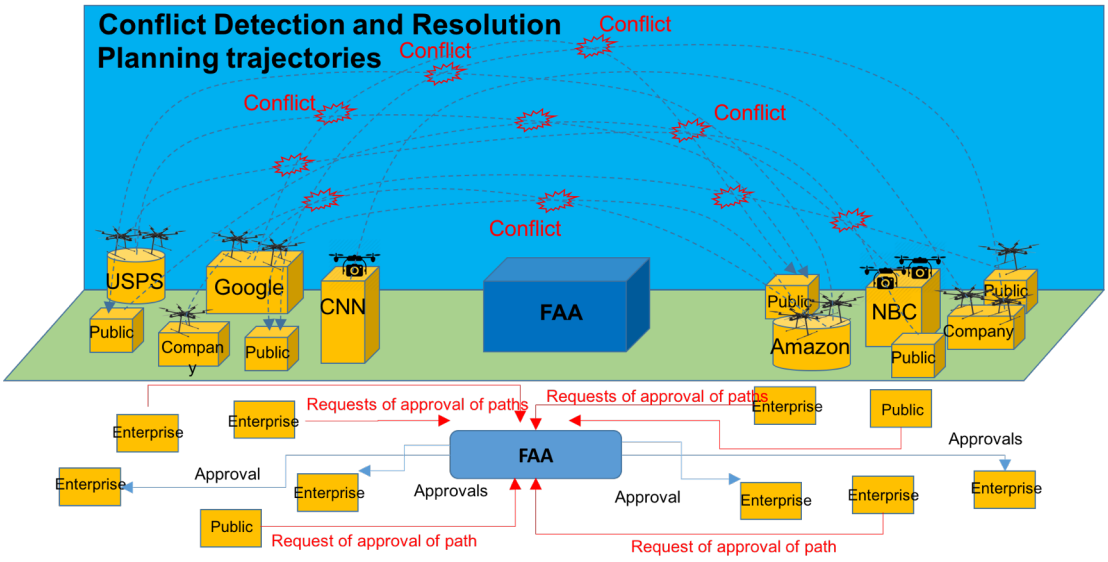


Figure 6.1: Air traffic congestion: conflict detection and resolution.

algorithms and protocols based on local sensors mounted on the drones can be employed to avoid collisions, but this can also slow the flow of the unmanned airspace traffic. To address the performance of unmanned airspace traffic management, we investigate centralized nominal flight plan negotiations between enterprises and the Federal Aviation Administration (FAA). The trajectory of each drone can be regarded as a sequence of waypoints, represented by a sequence of 4 dimensional data points involving x, y, z and time. Every nominal trajectory planned by a company must obtain initial approval from the FAA. If it has no conflict, the FAA may, but need not, approve it. There is a tradeoff between liberal regulation of the number of drones to increase capacity versus the probability of increased safety-induced aborts during real-time operation due to traffic congestion experienced in the sky.

The drones themselves will need to autonomously avoid collisions in real time, since wind and other interruptions can lead to large deviations from nominal flight paths. They will need to take into account factors such as their relative distances, velocities and other uncertainties like wind into account, in real-time. But, the flights of drones following trajectories which have potential collisions with others can not only threaten the safety of drones, but can also deteriorate the throughput of the air traffic by making drones keep changing their paths for real-time collision avoidance [60]. If the nominal trajectories requested for approval by the FAA have high possibilities of collisions, they may be denied. If not, they can be approved in order to improve the throughput performance of air traffic. We investigate the development of probability-based collision detection and resolution algorithms. We also conduct studies of overall system performance. Through

this process we address the improved performance of unmanned aerial system traffic management.

6.1. Problem Definition

We illustrate the problem and our solution on a prototypical situation arising in the collision resolution among planned trajectories of drones in a UTM system. Typically, the reference trajectory of the drone *i* is represented by

$$r^i = \{r_k^i = (x_k^i, y_k^i, z_k^i, t_k^i) | k = 0,1, \dots, K^i\}, \quad (6.1)$$

where (x_k^i, y_k^i, z_k^i) is a drone's 3-dimensional location at time t_k^i, and K^i is the number of reference positions. The real position of the drone *i* is described by

$$p_k^i = (x_k^i, y_k^i, z_k^i, t_k^i). \quad (6.2)$$

We assume that a drone flies between two reference positions at a constant speed

$$v_k^i = \frac{\sqrt{(x_{k+1}^i - x_k^i)^2 + (y_{k+1}^i - y_k^i)^2 + (z_{k+1}^i - z_k^i)^2}}{t_{k+1}^i - t_k^i}, \quad (6.3)$$

Therefore, we find its position at any arbitrary time t by linear interpolation:

$$(x^i(t), y^i(t), z^i(t)) = (x_k^i, y_k^i, z_k^i) + \lambda^i(t) \cdot \left((x_{k+1}^i, y_{k+1}^i, z_{k+1}^i) - (x_k^i, y_k^i, z_k^i)\right), \quad (6.4)$$

where $\lambda^i(t) = (t - t_k^i)/(t_{k+1}^i - t_k^i)$ for $t_k^i \le t \le t_{k+1}^i$.

Let the radii of two drones *i* and *j* be R_i and R_j. With $d^{ij}(t)$ representing the distance between the positions of the drones, a collision occurs between two drones when $d^{ij}(t) \le R_i + R_j$.

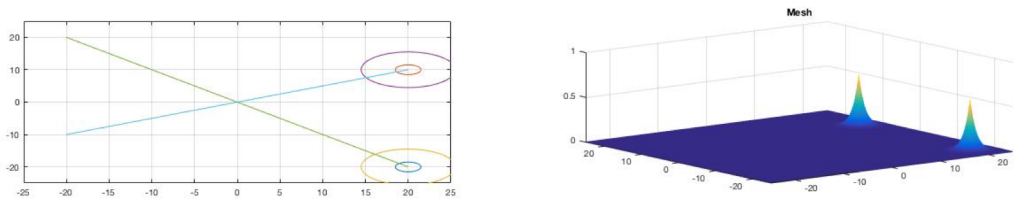
6.2. Probability-based Collision Detection and Resolution

The position of a drone flying outdoor is subject to external forces like wind. Since the exact position at each waypoint of its trajectory is unknown, we represent the position

of a drone can be represented by real valued random variables. The Gaussian distribution is used to describe the probability distribution of the position of the drone at any point along its trajectory. Given a waypoint of the trajectory, the probability density function of the position of the drone is assumed to be

$$\Pr\left(p^i | r^i, \sigma^2\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{|p^i - r^i|^2}{2\sigma^2}\right), \tag{6.5}$$

where r^i is the waypoint of the drone's trajectory, and σ^2 is a variance in the position of the drone. The value of a variance is chosen in relation to the effect of the external forces like wind. For example, when a drone flies in an area with a strong wind, the deviation from its trajectory becomes large with high probability. Figure 6.2 (a) shows two drones and their paths. The radii of their physical bodies are shown by small circles and the safety regions for protection against collision by larger circles. Figure 6.2 (b) represents the probability distributions of two drones' positions in two dimensions.



(a) (b)

Figure 6.2: Two drones' paths (a) and probability distributions of their positions in x-y plane (b).

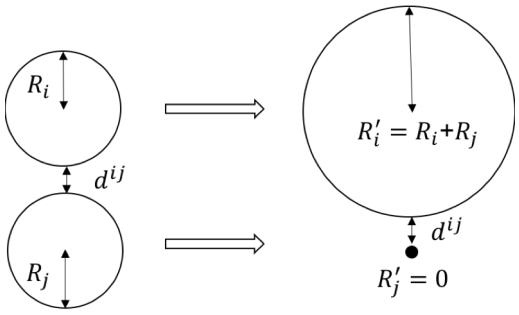


Figure 6.3: The simplified radii of drone *i* and *j*.

For simplification, we let the "effective" radius of drone *i* be the sum of the radii of drone *i* and *j*, and consider drone *j* as a point [62]. Figure 6.3 describes their effective radii. Moreover, with respect to their relative distance, we suppose that the position of the drone *i* is deterministic and simply assumed all the uncertainty is in the position of the other drone. The resulting probability density function of drone *j* has the following joint Gaussian distribution because their individual positions are independent and identically distributed (i.i.d.):

$$\Pr\left(\mathrm{p^j}|\mathrm{r^i}, \sigma_i, \mathrm{r^j}, \sigma_j, \mathrm{p^i}\ \right) = \frac{1}{2\pi\sigma_i\sigma_j}\exp\left(-\frac{{\sigma_j}^2\cdot\left|\mathrm{p^i}-\mathrm{r^i}\right|^2+{\sigma_i}^2\cdot\left|\mathrm{p^j}-\mathrm{r^j}\right|^2}{2{\sigma_i}^2{\sigma_j}^2}\right). \tag{6.6}$$

The position of drone *j* along its trajectory is parameterized:

$$\mathrm{r_k^j}(\lambda_\mathrm{k}) = \mathrm{r_0^j} + \lambda_\mathrm{k}\left(\mathrm{r_{K^j}^j} - \mathrm{r_0^j}\right), \tag{6.7}$$

where $0 \le \lambda_\mathrm{k} \le 1$, and $\mathrm{r_0^j}$and $\mathrm{r_{K^j}^j}$ are the starting and goal positions of drone *j*. Hence, when their distance is equal to the sum of their radii, we have

$$\begin{aligned}\left(\mathrm{R^i} + \mathrm{R^j}\right)^2 &= \left|\mathrm{r_{k^i}^i} - \mathrm{r_{k^j}^j}\right|^2 \\ &= \left|\mathrm{r_{k^i}^i} - \mathrm{r_0^j}\right|^2 - 2\cdot\lambda_{\mathrm{k^j}}\cdot\left(\mathrm{r_{k^i}^i} - \mathrm{r_0^j}\right)^\mathrm{T}\cdot\left(\mathrm{r_{K^j}^j} - \mathrm{r_0^j}\right) + {\lambda_{\mathrm{k^j}}}^2\cdot\left|\mathrm{r_{K^j}^j} - \mathrm{r_0^j}\right|^2, \end{aligned} \tag{6.8}$$

where $t^{i}_{k^i}=t^{j}_{k^j}$. We determine λ_{k^j} by solving the above for each segment of time $t^{j}_{k^j}$. The length of collision segment for λ_{k^j} is:

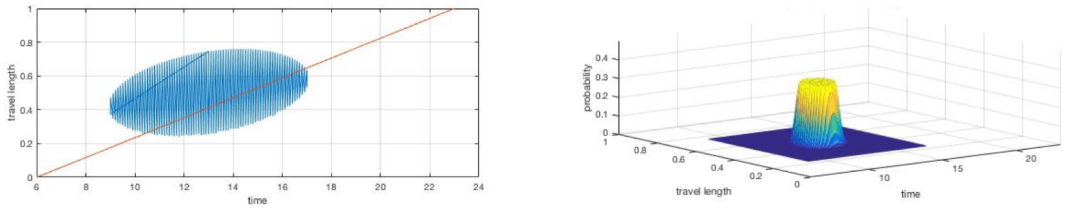
$$Length\ of\ collision\ segment = \begin{cases} \emptyset & if\ no\ real\ solution \\ (\lambda^{2}_{k^j} - \lambda^{1}_{k^j}) \cdot \left| r^{j}_{K^j} - r^{j}_{0} \right| & if\ two\ real\ solutions' \end{cases} \tag{6.9}$$

where $\lambda^{2}_{k^j} - \lambda^{1}_{k^j} \geq 0$.

Then, the probability density function along the collision segment is described by:

$$\Pr\left(\lambda | r^{i}_{k^i}, \sigma^2\right) = \frac{\alpha}{2\pi \cdot \sigma^2} \exp\left(- \frac{\left| r^{i}_{k^i} - r^{j}_{0} - \lambda\left(r^{j}_{K^j} - r^{j}_{0}\right) \right|^2}{2\sigma^2} \right), \tag{6.10}$$

where $\lambda^{1}_{k^j} \leq \lambda \leq \lambda^{2}_{k^j}$ and α is a parameter to normalize the probability distribution. The relationship between the trajectories of drone *i* and drone *j* can be described by their collision region, which shows the space-time uncertainty region occupied by drone *i* from the viewpoint of drone *j*'s trajectory [61]. Figure 6.4 (a) represents such collision region, indicated by the blue area with the trajectory of drone *j* indicated by the red line. The collision region is formed by the set of their collision segments. Figure 6.4 (b) illustrates the probability distribution of collision in the collision region.



(a) (b)

Figure 6.4: The collision region by drone *i* and the trajectory of drone *j* (a) and the probability distribution of the position of drone *i* on the collision region (b).

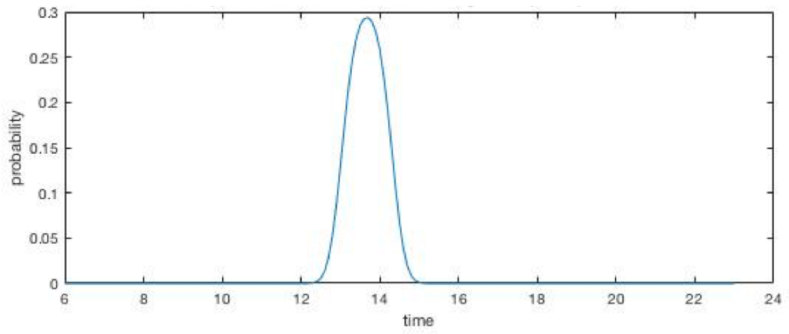


Figure 6.5: The probability distribution of the collision along the trajectory of the drone *j*.

Figure 6.5 depicts the probability distribution of collision along the trajectory when the trajectory of the drone *j*, shown by the red line, passes through the collision region in the Figure 6.4 (a).

The collision probability between drone *i* and drone *j* is calculated by integrating the probability density function of collision along the trajectory of drone *j* with respect to time.

$$\Pr\left(\left\lceil \mathrm{r}^{\mathrm{i}} - \mathrm{r}^{\mathrm{j}} \right\rceil \leq \mathrm{R}_{\mathrm{i}}' | \sigma^2\right) = \int_{\mathrm{t}_0^{\mathrm{j}}}^{\mathrm{t}_{\mathrm{f}}^{\mathrm{j}}} \frac{1}{2\pi \cdot \sigma^2} \exp\left(-\frac{(|\mathrm{r}^{\mathrm{i}} - \mathrm{r}^{j}| - \mathrm{R}_{\mathrm{i}}')^2}{2\sigma^2}\right) dt, \tag{6.11}$$

This collision probability between two drones can be reduced by changing the starting time of either of their trajectories. Park and et. al suggest assigning priorities to each trajectory and when a collision among two trajectories is detected, the starting time of the trajectory with a lower priority is delayed to avoid a collision [61][62]. However, such priority-based method can cause the performance of the entire system to be deteriorated by huge delays of the trajectories with low priorities, because they have to

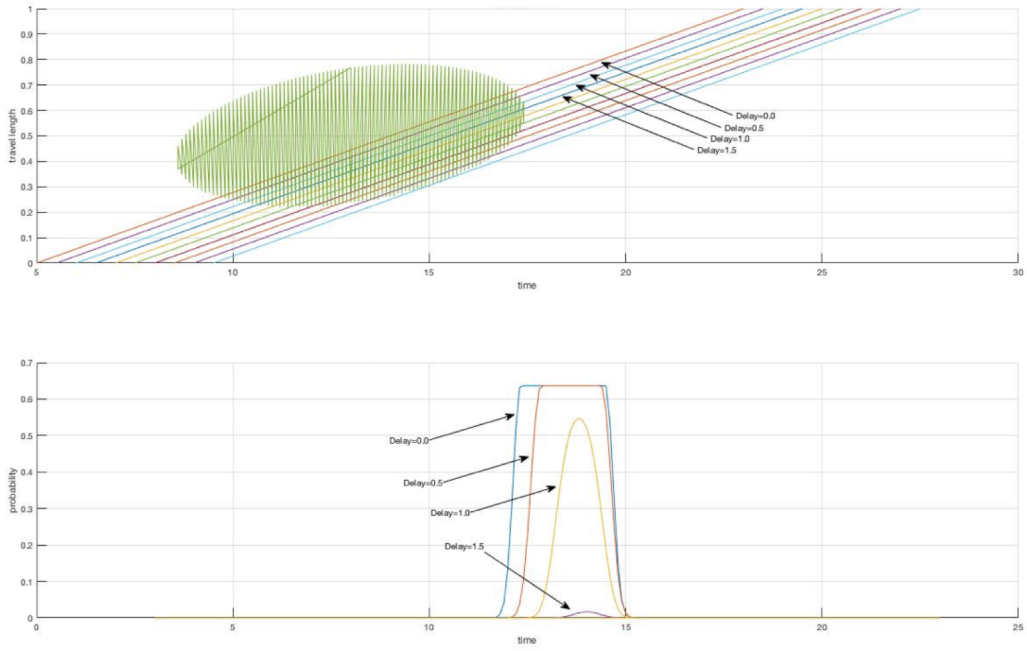


Figure 6.6: The variation of the probability distribution of collision according to the delays of the starting time of drone *j*.

wait until all the higher priority drones complete passage through the collision regions. In order to solve this problem, we suggest a probability-based gradient descent method for multiple trajectories to find the set of delays in starting time which reduces the collision probability among all trajectories to be below a certain threshold. As Figure 6.6 shows, the probability diminishes as the starting time of drone *j* is delayed, the probability distribution diminishes.

Algorithm 1 specifies how to find the delayed starting times which make the probability of collision among all trajectories less than a certain threshold. The fundamental sequence is as follows: Given the N number of trajectories, at each iteration we choose one of them and build a probability-based collision map related to others. Then,

Algorithm 1: Probability-based collision resolution.

procedure FindDelays(T, ε)
$N \leftarrow$ Number of trajectories in T
while ($! \overset{\forall}{\square} Pr_{min} \leq \varepsilon$) **do**
  for $i = 1 : i \leq N$ **do**
    $\tau \leftarrow (\tau^i - \Delta\tau, \tau^i, \tau^i + \Delta\tau)$
    $r^i(\tau) \leftarrow \{r_k^i = (x_k^i, y_k^i, z_k^i, t_k^i + \tau) | k = 0,1, \dots, K^i\} \in T$
    $\tau^i \leftarrow \arg\min_{\tau} \sum_{j=1, j\neq i}^{N} Pr(\lceil r^i(\tau) - r^j \rceil \leq R_i')$
    $Pr_{min}^i \leftarrow \sum_{j=1, j\neq i}^{N} Pr(\lceil r^i(\tau^i) - r^j \rceil \leq R_i')$
  end for
end while

we calculate the probability of collision for each starting time: the current, delayed and earlier starting times. The starting time of the current trajectory is updated as the starting time which has the minimum probability of collision. The step size of the starting time delay is one of the design parameters. A larger step size makes the convergence faster, but, it can also cause unnecessarily large deviations. On the other hand, a small step size helps to find minimum deviations, but, increases convergence time.

6.3. Simulation Results

We demonstrate through a simulation study the performance of the proposed algorithm ineffectively find the set of collision-free trajectories. We use MATLAB to develop the simulator. The computer used is 2.2GHz Intel Core i7 CPU with 4 GB DDR3 Memory. In the simulation, the flying space is 50 x 50 x 50 units, the average flight distance 86 units, and the average flight time 20 units. The initial positions and goal

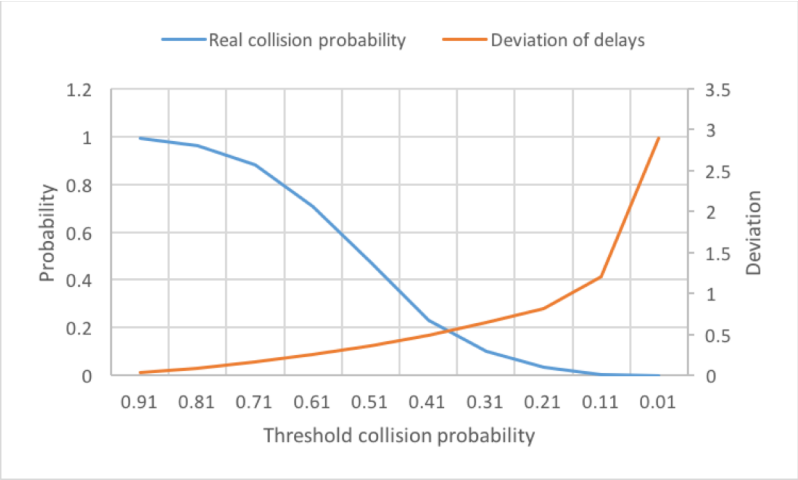


Figure 6.7: The variations of the real collision probability and deviation of delays of starting time according to the desired collision probability.

positions of the 20 trajectories are randomly generated. For simplicity, all trajectories are straight lines from their initial positions to goal positions. There is no priority among the trajectories. The proposed algorithm increases or decreases the starting times of trajectories at every iteration until the maximum probability of collision for all trajectories is less than the desired value.

Though it is important to reduce the collision probability for the safety of flight, it also entails a drop in the performance of an entire traffic control system. Figure 6.7 shows the variation of the deviation of the starting times and the real probability of collision with decreasing threshold probability for safety. Three 3D trajectories are used and the drones' real positions are determined by the Gaussian distribution with the mean as the waypoint of the trajectory, and the variance of value 2. The simulation is repeated 1000 times to obtain the probability of collision. As the desired probability of collision decreases, the probability of collision declines. However, it results in the deviation of the delayed starting

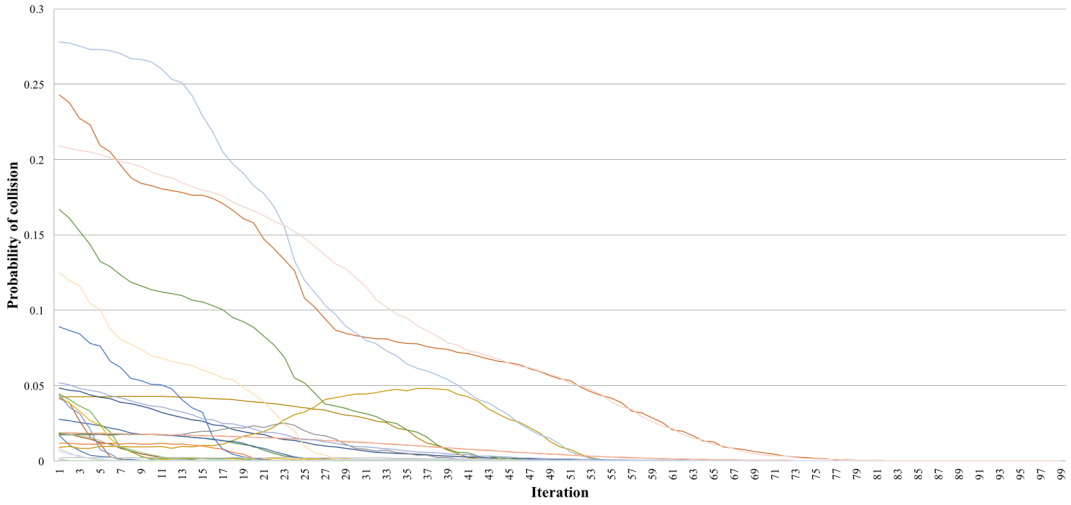


Figure 6.8: The convergence of the collision probabilities for 50 trajectories by the probability-based collision resolution algorithm

times increasing so that the performance of the traffic control system becomes progressively worse.

We validate the proposed algorithm by finding a set of collision-free starting times for 50 randomly generated trajectories. The parameter settings for the simulation are as follows: the step size for gradient descent is 0.1, the radius of a drone is 2, the radius of the safety region 0.5, and the variance of the drone position is 1. Figure 6.8 shows the variation of the collision probability of each trajectory at each iteration. The starting times are varied to reduce each trajectory's collision probability repeatedly so that all the collision probabilities of 50 trajectories converge to near zero.

We validate the proposed algorithm by comparing with a priority-based collision resolution algorithm. We generate 50 trajectories three times and compare their performances. Table 6.1 shows the results of the comparison between the two algorithms. When the 50 drones follow their unmodified initial trajectories, they collide 12 times, 12

Table 6.1: The comparison of the performances between the probability-based collision resolution and a priority-based collision resolution.

Algorithm	**Case 1**		**Case 2**		**Case 3**	
	Average square of delays	***Number of collisions***	***Average square of delays***	***Number of collisions***	***Average square of delays***	***Number of collisions***
Probability-based collision resolution	27.7048	0	22.5698	0	11.9416	2
Probability-based collision resolution	148.1359	4	57.8682	4	119.1597	6
No collision resolution	0	12	0	12	0	16

times and 16 times, respectively. The priority-based collision resolution algorithm lowers the number of collisions to 4 times, 4 times, and 6 times, respectively. However, due to the long delayed starting times of low-priority trajectories, the mean-squared delay is huge. This means that the priority-based algorithm while guaranteeing the safety among the trajectories, degrades the performance of the traffic system. On the other hand, the proposed probability-based collision resolution algorithm finds almost collision-free trajectories (0, 0, and 2 times, respectively) without increasing delay times much.

7. CONCLUSIONS

We have addressed the problem of cross-layer design of the cyber and physical layers, using a bi-directional middleware for cyber-physical systems that is an extension of Etherware, a prior separation-based middleware. The cross-layer design approach uses a Resource Manager which is developed to manage the interaction between the cyber layer and the physical layer, and resolve conflicts between them for resource usage. The first experimental case study illustrates how to handle the problem of network connectivity, a concern both for the cyber layer as well as formation control in the physical layer. Next, we have proposed a flexible adaptive model identification algorithm with outlier rejection for networked control systems. The performance of any model-based control system is strongly influenced by the accuracy of the system model. Hence, adaptive model predictive control is used to reduce the model error. However, the adapted model can be poor because the measurements or estimates used to adapt the model contain outliers due to the uncertainties of the communication network in a large-scale networked control system. This can degrade the reliability and robustness of the overall system. The proposed algorithms are incorporated using the Filter mechanism of Etherware so that they can be inserted into an existing MPC at runtime without additional computational burden and reconfiguration. We have experimentally demonstrated the improved performance obtained by the suggested overall system in a vehicular control testbed.

Two advanced applications of cyber-physical systems have also been illustrated. We have addressed the problem of cybersecurity of intelligent transportation systems. We have extended the theory of dynamic watermarking for linear systems to encompass

nonlinear systems describing a car's motion dynamics. When a malicious sensor node manipulates the position data of an autonomous car to deceive a controller node and thereby compromise the collision avoidance functionality, a dynamic watermarking mechanism detects the contamination of the sensing data and halts cars to prevent any collisions. We have implemented dynamic watermarking in the laboratory testbed comprised of a multiple vehicle control system, and have demonstrated that it effectively secures the safety of the intelligent transportation system in the presence of malicious sensor attacks. The second advanced application is an automated laser origami system for cyber-manufacturing. We have proposed a feedback-based laser origami control system for cyber-manufacturing. We have established a sensing system and a control system around a laser system to operate the laser cutting and bending processes automatically to make 3D objects. We also integrated the robotic arm with a laser control system to switch between cutting and bending processes automatically due to their different focal distances. We experimentally demonstrate that our proposed automatic laser system successfully produces a 3D creation by cut-bend operations with short processing time and high quality of bending.

Last, we have addressed a large scale distributed cyber-physical system comprised of automated drones operating in the 200ft-500ft above ground level portion of the Class G airspace. It is necessary to deconflict trajectories to the extent possible at the planning stage itself, in order to not induce too many mission aborts at run time during actual operation.

In an effort to solve this problem, we have proposed a probability-based collision resolution algorithm to find a set of collision-free trajectories from the offline planned trajectories. The proposed algorithm calculates collision probabilities among trajectories and modifies their starting times to reduce their collision probabilities to be less than a certain specified value to guarantee the safety of the entire system. Moreover, since the proposed algorithm modifies the starting times of trajectories in a round-robin manner, it does not degrade the performance of an air traffic with long-delays in starting times for any nodes, which is the problem with a priority-based collision resolution algorithm.

We have validated the proposed algorithm through simulation testing. We generated 50 trajectories randomly and compared the performance of the proposed algorithm with a priority-based collision resolution algorithm. We have demonstrated that the proposed algorithm guarantees better safety with smaller modifications of starting times than the priority-based collision resolution algorithm.

REFERENCES

[1] G. Baliga and P. R. Kumar, "A Middleware Architecture for Federated Control Systems," *IEEE Distributed Systems online*, Rio de Janeiro, Brazil, June 16-20, 2003.

[2] S. Graham and P.R. Kumar, "The Convergence of Control, Communication, and Computation," *Personal Wireless Communications Lecture Notes in Computer Science*, vol. 2775, pp. 458-475, 2003.

[3] K.D. Kim and P. R. Kumar, "Cyber-Physical Systems: A Perspective at the Centennial," *Proc. of the IEEE,* vol. 100, pp. 1287-1308, 2012.

[4] K.D. Kim and P. R. Kumar, "The Importance, Design and Implementation of a Middleware for Networked Control System," *Networked Control Systems Lecture Notes in Control and Information Sciences*, vol. 406, pp 1-29.

[5] M. Trivellato and N. Benvenuto, "Cross-Layer Design of Networked Control Systems", Proc. of IEEE ICC, 2009

[6] M.S. Branicky, S.M. Phillips, and Wei Zhang, "Scheduling and Feedback Co-Design for Networked Control Systems", Proc. of the 41st IEEE CDC, Dec. 2002

[7] N. Michael, M.M. Zavlanos, V. Kumar, and G.J. Pappas, "Maintaining Connectivity in Mobile Robot Networks", Springer Tracks in Advanced Robotics, Vol. 54, pp. 117-126, 2009

[8] D. Soudbakhsh, L. T.X. Phan, O. Sokolsky, I. Lee and A. Annaswamy, "Co-design of Control and Platform with Dropped Signals", ICCPS, Philadelphia, April, 8-11, 2013.

[9] D. Sauter, M. A. Sid, S. Aberkane and D. Maquin, "Co-design of safe networked control systems", Elsevier, Annual reviews in control, vol. 37, issue 2, pp. 321-332, Dec. 2013.

[10] C. L. Robinson, H.-J. Schutz, G. Baliga and P. R. Kumar, "Architecture and Algorithm for a Laboratory Vehicle Collision Avoidance System," *IEEE ISIC*, Singapore, Oct. 1-3, 2007.

[11] H.J. La and S.D. Kim, "A Service-based Approach to Designing Cyber Physical Systems*," IEEE 9th ACIS ICCIS*, pp. 895-900, Aug. 18-20, 2010.

[12] Y. Wang, G. Tan, Y. Wang and Y. Yin, "Perceptual control architecture for cyber-physical systems in traffic incident management," *Journal of Systems Architecture*, vol. 58, pp. 398-411, 2011.

[13] T. Sanislav and L. Miclea, "Cyber-Physical Systems – Concept, Challenges and Research Areas," *CEAI*, vol. 14, 2012, no. 2, pp. 28-33.

[14] E. A. Lee, "Cyber Physical Systems: Design Challenges," *ISORC, 2008 11th IEEE International Symposium*, pp. 363-369, May 5-7, 2008.

[15] K. Wan, D. Hughes, K.L. Man, and T. Krilavicius, "Composition Challenges and Approaches for Cyber Physical Systems," *IEEE International Conf. on NESEA*, 25-26 Nov. 2010.

[16] A. Rajhans, S.-W. Cheng, B. Schmerl, D. Garlan and B.H. Krogh, "An Architectural Approach to the Design and Analysis of Cyber-Physical systems," *Proceedings of the 3rd International Workshop on MPM,* 2009.

[17] C.G. Kim, M. Sun, S. Mohan, H.C. Yun, L. Sha, and T.F. Abdelzaher, "A Framework for the Safe Interoperability of Medical Devices in the Presence of Network Failures," *ICCPS*, April 13-15, 2010, Stockholm, Sweden.

[18] Z. Wang and H.T. Sun, "Control and Scheduling Co-design of Networked Control System: Overview and Directions", Proc. of the ICMLC, Xian, 15-17 July, 2012.

[19] M.G. Forbes, R.S. Patwardhan, H. Hamadah and R.B. Gopaluni, "Model Predictive Control in Industry: Challenges and Opportunities", in ISACCP, Whistler, June 7-10, 2015.

[20] K.D. Kim and P.R. Kumar, "An MPC-Based Approach to Provable System-Wide Safety and Liveness of Autonomous Ground Traffic", IEEE Transactions on Automatic Control, Special Issue on "Control of Cyber-Physical Systems." Vol. 59, no. 12, pp. 3341-3356, December 2014.

[21] M. Gupta, K.R. Krishnanand, H.D. Chinh, and S.K. Panda, "Outlier Detection and Data Filtering for Wireless Sensor and Actuator Networks in Building Environment", IEEE ICBEST, Singapore, pp. 95-100, Aug. 31-Sep.1, 2015

[22] M. Takahashi, K Nonaka, and K. Sekiguchi, "Vehicle State Estimation by Moving Horizon Estimation considering Occlusion and Outlier on 3D Static Cameras", IEEE CCA, Sydney, Australia, pp. 1211-1216, Sep. 21-23, 2015

[23] B. Chen, B. Luan and K. Lee, "Design of Lane Keeping System Using Adaptive Model Predictive Control", in IC-CASE, Taipei, August 18-22, 2014.

[24] R.K. Pearson, “Outliers in Process Modeling and Identification”, IEEE Trans. on Control Systems Tech., vol., 10, no. 1, Jan., 2002.

[25] T. Kim, H. Fukushima and T. Sugie, “Robust Adaptive Model Predictive Control based on Comparison Model”, in IEEE CDC, Atlantis, December 14-17, 2004.

[26] K.D. Kim and P. R. Kumar, “A Real-Time Middleware for Networked Control Systems and Application to an Unstable System." IEEE Transactions on Control Systems Technology, vol. 25, no. 5, pp. 1898-1906, September 2013.

[27] W.Y. Han and J.C. Lin, “Minimum-maximum exclusive mean (MMEM) filter to remove impulse noise from highly corrupted images,” Electronics letters, 16th January, 1997, Vol.33, No.2, pp. 124-125.

[28] G. Baliga, S. Graham, S. Lui and P.R. Kumar, “Etherware: domainware for wireless control networks” Object-Oriented Real-Time Distributed Computing, Proc., 7th IEEE International Symposium, pp. 155-162, 2004.

[29] “Hackers remotely kill a jeep on the highway- with me in it.” [Online]. Available: https://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/

[30] C.Miller and C. Valasek, “Remote exploitation of an unaltered passenger vehicle,” Black Hat, USA, 2015

[31] “Hackers reveal nasty new car attacks-with me behind the wheel (video).” [Online]. Available: http://www.forbes.com/sites/andygreenberg/2013/07/24/hackers-reveal-nasty-new-car-attacks-with-me-behind-the-wheel-video/#29621f635bf2

[32] “Security experts say that hacking cars is easy.” [Online]. Available: http://fortune.com/2016/01/26/security-experts-hack-cars/

[33] “Your next car will be hacked. Will autonomous vehicles be worth it?” [Online]. Available: https://www.theguardian.com/technology/2016/mar/13/autonomous-cars-self-driving-hack-mikko-hypponen-sxsw

[34] Y. Mo and B. Sinopoli, “Secure control against replay attacks,” in 47th Annual Allerton Conference on Communication, Control, and Computing, Sept 2009.

[35] Y.Mo, S.Weerakkody, and B.Sinopoli, “Physical authentication of control systems: Designing watermarked control inputs to detect counterfeit sensor outputs,” IEEE Control Systems, vol. 35, no. 1, pp. 93–109, Feb 2015.

[36] S. Weerakkody, Y. Mo, and B. Sinopoli, “Detecting integrity attacks on control systems using robust physical watermarking,” in 53rd IEEE Conference on Decision and Control, Dec 2014, pp. 3757–3764.

[37] B. Satchidanandan and P. R. Kumar, “Dynamic watermarking: Active defense of networked cyber-physical systems,” Proceedings of the IEEE, to appear.

[38] B. Satchidanandan and P. R. Kumar, “Secure control of networked cyber-physical systems,” in 55th IEEE Conference on Decision and Control, 2016.

[39] G. Ambrogio, L. D. Napoli, L. Filice, F. Gagliardi and M. Muzzupappa, "Application of Incremental Forming process for high customised medical product manufacturing," Journal of Materials Processing Technology, vol. 162, pp. 156-162, 2005

[40] Z. Hu, R. Kovacevic and M. Labudovic, " Experimental and numerical modeling of buckling instability of laser sheet forming of laser sheet forming," International Journal of Machine Tools and Manufacture, vol. 42 , no. 13, pp. 1427-1439, 2002.

[41] J. M. Allwood, O. Music, A. Raithathna and S. R. Duncan, "Closed-loop feedback control of product properties in flexible metal forming processes with mobile tools," CIRP Annals-Manufacturing Technology, vol. 58, no. 1, pp. 287-290, 2009.

[42] W. Ko, B. Satchidanandan and P. R. Kumar, "Theory and Implementation of Dynamic Watermarking for Cybersecurity of Advanced Transportation Systems", International Workshop on Cyber-Physical Systems Security (CPS-Sec), pp. 235-239, Philadelphia, October 17-19, 2016

[43] B. Satchidanandan and P. R. Kumar, "On Minimal Tests of Sensor Veracity for Dynamic Watermarking-Based Defense of Cyber-Physical Systems.", 9th International Conference on Communication Systems & Networks (COMSNETS 2017), Bengaluru, January 4-8, 2017

[44] G. Blaiga, S. Graham, L. Sha, and P. R. Kumar, "Service continuity in networked control using Etherware." IEEE Distributed Systems Online, 15 pages, vol. 5, No. 8, September 2004. ISSN: 1541-4922

[45] S. Graham, G. Baliga and P. R. Kumar, "Abstractions, Architecture, Mechanisms, and a Middleware for Networked Control." IEEE Transactions on Automatic Control, vol. 54, no. 7, pp. 1490-1503, July 2009.

[46] S. Graham, G. Baliga, and P. R. Kumar, “Issues in the Convergence of Control with Communication and Computing: Proliferation, Architecture, Design, Services, and Middleware," Proceedings of the 43rd IEEE Conference on Decision and Control, Paradise Island, Bahamas, pp. 1466-1471, December 14-17, 2004.

[47] P. Kopardekar, “Safely Enabling Low-Altitude Airspace Operations: Unmanned Aerial System Traffic Management (UTM)”, NASA Technical Reports Server, April 15, 2015

[48] P. Wright, “Cyber-physical product manufacturing”, Manufacturing Letters, 2014. 2(2): p. 49-53.

[49] M. M. Herterich, F. Uebernickel and W. Brenner, “The Impact of Cyber-Physical Systems on Industrial Services in Manufacturing”, Procedia CIRP, 2015, 30: pp. 323-328.

[50] J. Lee, B. Bagheri and H.A. Kao, “Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems”, Manufacturing Letters, 2015, 3(0): pp. 18-23.

[51] S. Mueller, B. Kruck and P. Baudisch, “LaserOrigami: Laser-Cutting 3D Objects”, CHI 2013, April 27-May 2, 2013, Paris, France, pp. 2586-2592.

[52] S. Mueller, LaserOrigami: laser-cutting 3d objects, 2013. Available from: https://www.youtube.com/watch?v=arjRtCjI9AQ.

[53] D. J. Balkcom and M. T. Mason, “Introducing robotic origami folding”, Proceedings on IEEE International Conference on Robotics and Automation, April 26 - May 1, 2004, New Orleans, USA

[54] A. Namiki and S. Yokosawa, “Robotic Origami Folding with Dynamic Motion Primitives”, IEEE /RSJ International Conference on Intelligent Robots and Systems, Sept 28 - Oct 2, 2015, Hamburg, Germany, pp. 5623-5628.

[55] K. Tanaka, Y. Kamotani and Y. Yokokohji, “Origami folding by a robotic hand”, IEEE/RSJ International Conference on Intelligent Robots and Systems, Oct. 29 - Nov. 2, 2007

[56] A. Eideh, U. S. Dixit and R. Echempati, “A Simple Analytical Model of Laser Bending”, 5th International and 26th All India Manufacturing Technology, Design and Research Conference, Dec. 12-14, 2014, IIT Guwahati, Assam, India

[57] Lasersaur, http://www.lasersaur.com/

[58] Dobot magician robotic arm, https://www.dobot.us/

[59] The automatic two-side laser bending, Available from: https://youtu.be/Ufeu0zgYzng

[60] Y. Lin and S. Saripalli, “Sampling Based Path Planning for UAV Collision Avoidance”, IEEE Transactions on Intelligent Transportation Systems, vol. PP, issue: 99, pp. 1-14, 25 April 2017

[61] S.H. Park and B.H. Lee, “Analysis of robot collision characteristics using the concept of the collision map”, Robotica, vol. 24, pp. 295-303, 2005

[62] S.H. Ji, J.S. Choi, and B.H. Lee, “A Computational Interactive Approach to Multi-agent Motion Planning”, International Journal of Control, Automation, and Systems, vol. 5, no. 3, pp. 295-306, June 2007

[63] P. H. Kopardekar, “Safety Enabling UAS Operations in Low-Altitude Airspace”, UTM Convention, Moffett Field, CA, Oct. 2015

[64] C. Wargo, C. Snipes, A. Roy and R. Kerczewski, “UAS Industry Growth: Forecasting Impact on Regional Infrastructure, Environment, and Economy”, IEEE-DASC, Sacramento, CA, 26-29 Sep. 2016

[65] M. Chen, Q. Hu, J. F. Fisac, K. Akametalu, C. Mackin and C. J. Tomlin, “Reachability-Based Safety and Goal Satisfaction of Unmanned Aerial Platoons on Air Highways”, Journal of Guidance, Control and Dynamics, 30 Jan. 2017

[66] T. Kistan, A. Gardi, R. Sabatini, S. Ramasamy and E. Batuwangala, “An evolutionary outlook of air traffic flow management techniques”, Progress in Aerospace Sciences, vol. 88, pp. 15-42, 14 Nov. 2016

[67] J.L. Rios, D.G. Mulfinger, I.S. Smith, P. Venkatesan, D.R. Smith, V. Baskaran and L. Wang, “UTM Data Working Group Demonstration 1 Final Report”, SGT Inc. April 2017, Moffett Field, CA

[68] F. Borrelli and D. Subramanian, “MILP and NLP Techniques for Centralized Trajectory Planning of Multiple Unmanned Air Vehicles”, Proceedings of the 2006 ACC, Minneapolis, Minnesota, USA, June 14-16, 2006

[69] D. Alejo, J.A. Cobano, G. Heredia and A. Ollero, “Collision-Free 4D Trajectory Planning in Unmanned Aerial Vehicles for Assembly and Structure Construction”, Journal of Intelligent Robot System, vol. 73, pp. 783-795, 2014