A Heterogeneous Fleet of Vehicles for Automated Humanitarian Missions

Pieter J. Mosterman [*] **David Escobar Sanabria** [**]
Enes Bilgin [***] **Kun Zhang** [****] **Justyna Zander** [†]

[*] *MathWorks, Natick, MA, USA (pieter.mosterman@mathworks.com)*
[**] *University of Minnesota, Minneapolis, MN, USA (descobar@umn.edu)*
[***] *Boston University, Boston, MA, USA (enes@bu.edu)*
[****] *University of Arizona, Tucson, AZ, USA (dabiezu@email.arizona.edu)*
[†] *HumanoidWay, Natick, MA, USA (dr.justyna.zander@ieee.org)*

Abstract:
Natural disasters are of all times and as technology becomes available its utility in disaster response and relief is exploited. This work presents an automated emergency response system and an experimental framework for its design and validation. Given a set of requests from the field and infrastructure information, a high-level optimization method generates a mission plan for a fleet of autonomous vehicles. The fleet includes ground vehicles for setting up local stations, fixed wing aircraft for assessing infrastructure damage, and rotorcraft for delivering emergency supplies. Internet technology provides a unifying environment for the vehicles, optimization module, operators, and emergency responders with support for computational integration in cyberspace. Experiments validate the guidance and control strategies for the rotorcraft vehicles and show the feasibility of the proposed system in a preliminary sense.

Keywords: Humanitarian, Cyber-Physical Systems, Operations, Virtual Integration, Modeling, Simulation, Wireless Control, Model-Based Design, Visualization

1. INTRODUCTION AND CONTEXT

As a recent report of the Intergovernmental Panel on Climate Change (IPCC Panel (2013)) confirmed, there is substantial evidence that humans are affecting and are being affected by global climate change. This change may well be responsible for intensifying effects of natural disasters such as storms, floods, earthquakes, and droughts which have an unequal impact on the world population. Those living in poor and developing countries have less of an ability to adapt. Yet, few humanitarian non-governmental organizations or international governmental organizations are prepared to address the implications of this inequality. With technology as a potential equalizer, this work explores requirements for humanitarian missions and the feasibility to address these with emerging technologies and the cyber-physical systems paradigm.

1.1 Humanitarian Mission Requirements

ScienceDaily (2005) reports a sharp world-wide increase in natural disasters, from around 100 per year in the early 1960s, to 500 to 800 per year in the early 21st century. The raise is not so much because of an increase of disastrous events, but more so because people have been spreading to vulnerable locations such as near the sea, in part conduced by global climate change. "We know how to prepare for disasters, but the world has not made this a high enough priority," Sarewitz said in ScienceDaily (2005). Reducing emissions is important, but will not reduce vulnerability to disasters. If disaster preparation received the same political attention as global warming, significant progress could be made." Still, the issues of climate change and disaster vulnerability remain separated in the eyes of the media, public, environmental activists, scientists, and policymakers.

Attempts to find solutions involve disciplines such as system engineering, computer science, physics, but also disaster management, health management, and policy making. Technology can help provide, collect, and store essential information, and share this information among people and organizations involved. Challenges in monitoring a population after a natural disaster result from the nature of an emergency system, which is substantially different from a day-to-day information system.

The effectiveness of emergency response depends on the quick provision of detailed, precise, and up-to-date information (e.g., Federal Emergency Management Agency (2010)). Not only the natural environment must be monitored, but also the state of traffic, hospitals, and civil infrastructures such as electricity and water supplies. In addition, the location, status, and number of injured people must be provided in a timely fashion (e.g., McEntire (2005)). The information infrastructure availability may also decrease in the face of a disaster and the state of relevant objects may change extremely quickly. For instance, hospitals may close down operations abruptly because of

electricity shortcuts, while emergency medical-care points start operations at unpredictable points in time (Careem et al. (2006)). Furthermore, humans play an essential role because of their unique ability to assess, interpret, and access information which makes visualization imperative to make huge amount of data easily accessible.

1.2 Challenges

In these times of nature exposing its dangers, evolving technologies provide an opportunity to mitigate the effect of dissasters. Embedded systems and network technology have come to tightly integrate cyberspace and physical devices referred to as *cyber-physical systems* (CPS) Lee (2008). The CPS paradigm captures the requirements of emergency response systems well because of the support for flexibility in system configuration. In particular, designing for *new and unknown* system functionalities that may become required or active not sooner than during deployment relates to emergency and challenges the traditional approach of system design (Mosterman and Zander (2014)). Thus, novel system-level engineering approaches become a necessity for disaster mitigation. New *open system architectures* and more powerful *design-time* approaches are necessary to conceptualize system behavior at various levels of detail in different parts of the CPS (Steering Committee for Foundations in Innovation for Cyber-Physical Systems (2013)).

This paper presents an experimental system where multiple vehicles of different types that interact in a (possibly modeled) physical world come together and are controlled and coordinated via cyberspace to accomplish a complex logistical operation. The physical world involves operating vehicles (e.g., cars, drones) with digital control on a *time scale of milliseconds* and confined to the dimensions of an individual vehicle. Furthermore, serving needs of an emergency responder puts *humans squarely in the loop* with the according challenges such as the manner in which humans and machines best interact, safety aspects, and security. Cyberspace includes optimization-based planning methods that operate at *time scales in the order of minutes* at a possibly global network level.

The proposed testbed provides an opportunity to study implementation feasibility with current design methods and to assess gaps and pitfalls when extending the system with *increasingly complex collaboration* between vehicles. The system responds to requests and information about the state of the infrastructure; for example, to reroute the *already deployed* fleet of vehicles in a *collaborative fashion*. Further, an *open architecture* allows adding vehicles (e.g., sensor drones) to the fleet anytime during design or deployment. Moreover, functionality that is shared by a collaborating fleet migrates from local to remote locations (provided by *cloud-based services*) when the vehicle routes and operations are reassigned.

Previous work has addressed parts of the system presented herein. For example, Cassandras and Paschalidis (2011) used cooperative receding horizon methodology to plan out vehicle trajectories over a limited time window. Yin et al. (2012) studied use of social media to attain situational awareness during emergencies while Beetz (2005) designed plan-based control for autonomous robots. The RoboEarth network is an interesting related attempt described by Waibel et al. (2011) where worldwide robots connect to generate, share, and reuse data. Related in another dimension is the humanoid robotic emergency responder effort by Honda (Falconer (2013)).

This paper puts forward an attempt at a comprehensive system to facilitate humanitarian missions. To this end, Section 2 proposes an automated and autonomous emergency response system. In Section 3, a system implementation is described. Section 4 provides details on the control and guidance strategies for the autonomous vehicles. Section 5 presents conclusions and an outlook.

2. AN AUTOMATED AUTONOMOUS EMERGENCY RESPONSE SYSTEM

Autonomous vehicles collaborate in a layered planning and control structure of the emergency response system.

2.1 Cyber-Physical Systems for Emergency Response

Developments in the area of *cyber-physical systems* are making an automated emergency response system a realistic vision. A humanitarian project at MathWorks studied the opportunity for automation in responding to the aftermath of a natural disaster such as an earthquake. The overall goal is to automate delivery of emergency supplies in the field. During a humanitarian action, dynamic requests for help are sent to the central mission system. Stations should serve as depots and deployed to locations so that upon requests supplies can be transported in minimum time. In addition, reconnaissance of the disaster area is necessary to determine the availability of the road network and to make an overall damage assessment, which helps predict the density of future requests.

Figure 1 shows the conceived system. In the center at the bottom, the mission system is shown, comprising a user interface and a mission planning component.

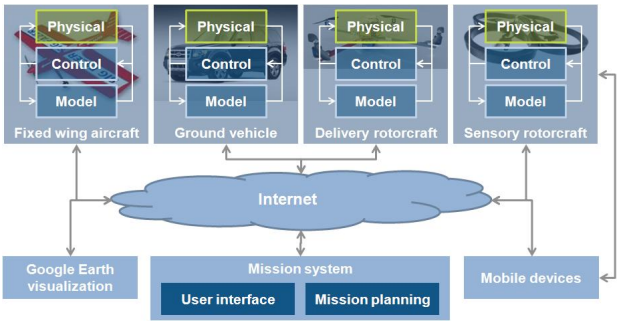


Fig. 1. Emergency response system

Different types of autonomous vehicles serve the requests: (i) fixed wing aircraft fly reconnaissance sorties, (ii) ground vehicles set up local deployment stations, and (iii) quadcopter rotorcraft deliver supplies and tools. Once a mission plan is generated, each of the individual vehicles in the fleet is sent their set of waypoints to follow, including dwell time at each waypoint. In the experimental system, control is applied to either the physical vehicles or their models. A soft real-time Google Earth visualization integrates the fleet of vehicles (physical or modeled) in a computational

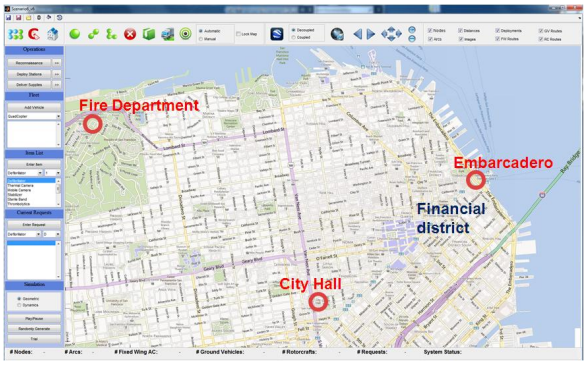


Fig. 2. The mission user interface for San Francisco, CA

sense. With a mobile device, an emergency responder can seize control of an autonomous sensory drone to accurately and reliably home in and land it.

The project classifies emergency response by requests for: (i) damage assessment, (ii) supplies (e.g., defibrillators, masks, thermal blankets), (iii) tools (e.g., CO_2 analyzer, camera, thermal imaging sensor), and (iv) surveillance. Either emergency responders in the field or 911 operators enter requests (including geographic location in terms of longitude and latitude) into the mission coordination system. A prepopulated list of request types consists of: defibrillator, thermal camera, mobile camera, stabilizer, sterile band, thrombolytics, thermal blanket, carbamazepine, oral airway, endotracheal tube, and mask.

Given the requests, the available infrastructure, and a characterization of the various vehicles, solving a multi-objective optimization problem generates the locations for each of the vehicles to visit. Based on the infrastructure information, trajectories between the locations are generated as sets of waypoints. For example, an earthquake scenario in San Francisco may involve a large number of delivery requests from the financial district. Optimized deployment may determine that the ground vehicles should set up station at the Embarcadero and city hall (see Fig. 2). To drive from the fire department to these locations, trajectories of waypoints along the available roads are generated for each of the ground vehicles. To this end, a predetermined set of midpoints along the roads of San Francisco is available from government provided public 'shape files'.

Based on the ground station deployment, the optimization problem solves the dispatch of delivery drones from the stations. The trajectories of waypoints for the delivery rotorcraft are generated following the roads to avoid having to account for the height of buildings. Likewise, trajectories are generated for the fixed wing aircraft based on user specified locations for reconnaissance and based on the trajectories generated for the ground vehicles to determine the state of the infrastructure for a planned route.

In a typical experiment involving computational resources and vehicle simulations, an emergency response scenario would consist of 3 to 4 fixed wing aircraft, two of which are on a reconnaissance mission, 4 to 5 ground vehicles, and 15 to 20 delivery drones. Experimenting with the implementation showed the potential for automatically

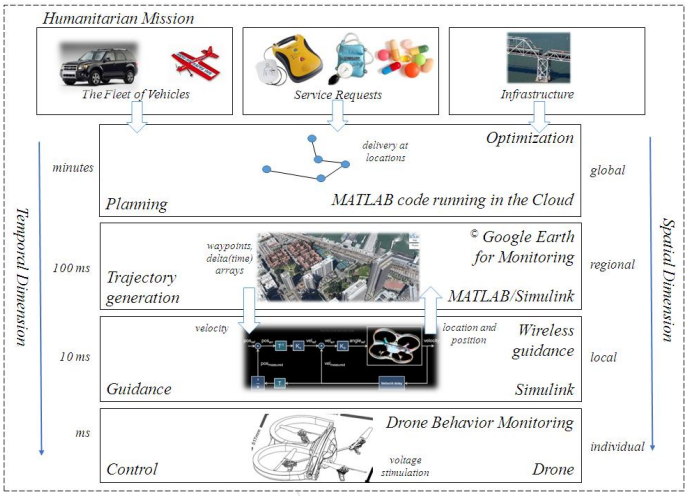


Fig. 3. Planning and control layers of the overall system

deploying autonomous vehicles to serve a range of 80 to 100 requests from the field in a time optimal manner.

Kress-Gazit et al. (2009) tackles a similar problem by exploring temporal logic as planning input. While this allows a rich vocabulary to express mission constraints, it renders generation of an optimal plan for a heterogeneous fleet more complex. In the current implementation such richness beyond 'next' and 'after' is gratuitous, and, therefore, omitted.

2.2 Characterization of the Control Layers

The planning, guidance, and control in the overall system spans a range of temporal scales as illustrated in Fig. 3. First, to set up the mission parameters: configure the fleet of vehicles, obtain the delivery and reconnaissance requests, and enter the available infrastructure. Next, a time-optimal mission plan with locations for each of the vehicles to visit is computed, which takes in the order of minutes. This plan spans a range of kilometers while the computation may occur on a global scale, for example, to take advantage of server farms.

The next stage generates waypoints along the roads with regional computing facilities, which takes in the order of seconds. The distance between pairs of consecutive locations is roughly of the order of 10-100 meters.

One level below, the motion control to move from one waypoint to the next operates as closed-loop feedback control on a temporal scale in the order of 100 milliseconds. For example, for rotorcraft roll and pitch angles may be controled to achieve a forward velocity. Motion control may be implemented on the computational resources of the vehicle or networked control may be applied, which could be executing locally (e.g., the ground vehicle station may control the motion of a dispatched drone). The distance between consecutive waypoints is approximately in the order of 1-10 meters.

At the lowest level is the actuator control, operating at a temporal scale of milliseconds. For example, this pertains to computing the control voltage of the electrical motors that determine the rotor speed of a rotorcraft. The actuator control is local to a vehicle which is about of

Vehicle	Model	Payload	Operational time	Speed
Ground vehicle	Ford Escape	500 kg	inf	0-25 m/s
Fixed wing aircraft	FASER	2.5 kg	25 min	7-35 m/s
Delivery rotorcraft	Arducopter	0.4 kg	9 min	0-14 m/s
Sensory rotorcraft	AR.Drone 2.0	250 g	12 min	0-5 m/s

Table 1. Characterization of the fleet vehicles

the order of 10-100 cm. The performance constraints on actuator control mandate that it could be executed on on-target computing resources.

3. IMPLEMENTATION OF THE SYSTEM

Architectural elements of the implementation and the communication between them is captured next.

3.1 Architecture

Figure 4 shows the architecture of the experimental system with concurrently operating resources in dashed outline. In a MATLAB R2013b (MathWorks® (2013a)) process, the overall mission control system at the bottom executes: (i) the mission user interface, (ii) the planning to generate a specific mission, (iii) the visualization module of the plan while it is being generated and the final result, and (iv) an animation of the mission progress during operations.

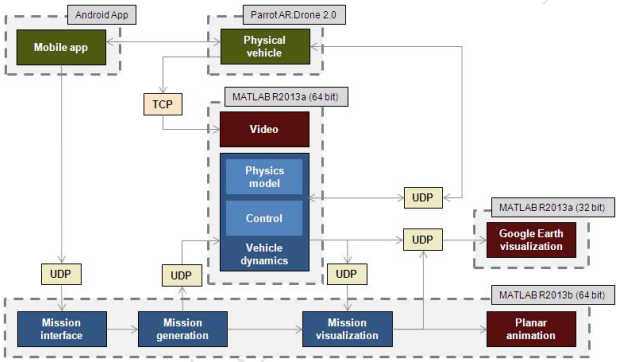


Fig. 4. System architecture

The MATLAB 2013a (MathWorks® (2013b)) process in the center of Fig. 4 generates the dynamics for the fleet vehicles (see Table 1) and includes the waypoint control. For the ground vehicles and fixed wing aircraft, the control is applied to a vehicle model. For rotorcraft the control is either applied to a model or the physical vehicle, the latter of which operates as a concurrent resource. In addition to vehicle state information, the physical rotorcraft is capable of providing a video stream during operation.

The vehicle state information is provided to a MATLAB 2013a (MathWorks® (2013b)) process that must be a 32 bit version for Google Earth three dimensional visualization. Further, the mission system provides configuration information to Google Earth such as which vehicle viewpoint should be displayed and how this viewpoint should be displayed (e.g., from the vehicle perspective, from a trailing camera, etc.).

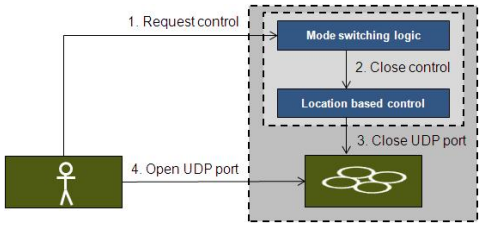


Fig. 5. Seizing control of the drone

Finally, the top left-hand corner of Fig. 4 shows a process on a mobile device that executes an application to directly enter requests into the mission system. The Android automatically attaches longitude and latitude coordinates to a request.

3.2 The Network Connections

Wired and wireless networks connect the resources in Fig. 4. UDP port objects in MATLAB or UDP blocks in Simulink communicate across the internet between the MATLAB processes and with the Android application. The MATLAB processes communicate with the physical sensory drone via its *ad hoc* wireless network or an available wireless netowork. UDP blocks in Simulink write control values to the drone IP address and port while reading state values from the drone in a similar fashion. This allows wireless control of the physical sensory drone directly from Simulink using Real-Time Windows Target for real-time performance (MathWorks® (2013b)). The video stream is communicated to MATLAB or Simulink by a TCP protocol based connection.

Figure 5 shows how an emergency responder seizes control of a sensory drone. First, the drone is requested to hand over control via a mobile device application connected to the internet. The autonomous drone guidance receives this request via a UDP connection and closes the UDP connection to the drone control. The mobile device then opens a UDP port and starts sending control commands. In the experimental implementation, control was transferred while the drone was hovering in mid air, resulting in a small drop in height (approximately 50 cm, Fig. 6).

(a) Autonomous control to hover in place (b) Drop when changing control (c) Responder controlling the drone

Fig. 6. Change of control (post processed drone outline)

4. AUTONOMOUS FLEET OF VEHICLES

Section 4.1 presents an overview of the dynamics and control and guidance strategies for the vehicles of the emergency reponse system. The Model-Based Design of control and guidance systems, applied to the sensory rotorcraft, is illustrated in Section 4.2.

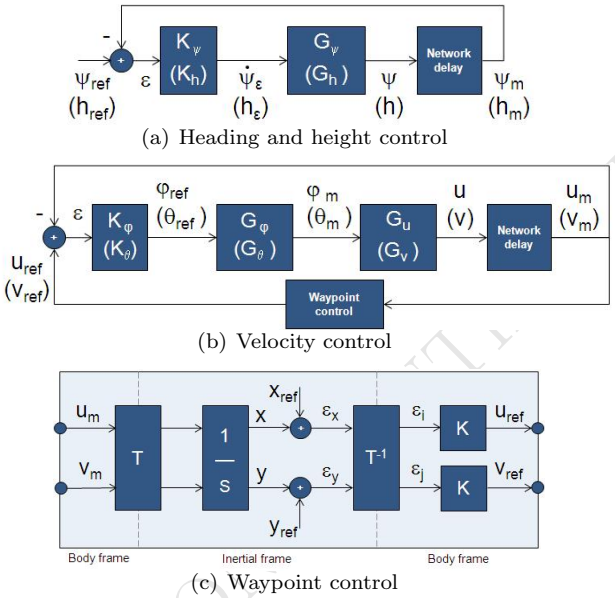
4.1 Dynamics, Control, and Guidance

A discussion on the modeling, low level control, and experimental validation of the fixed wing aircraft is presented

in (Dorobantu et al. (2013)). The vehicle orientation and velocity are driven by an inner-loop controller using the elevator, rudder, throttle, and ailerons (left and right). Height is controlled using the pitch angle command. Turns are generated using the roll angle, which varies the heading rate of the aircraft.

A dynamics model of the AR.Drone sensory rotorcraft is derived using experimental data, structure of the vehicle equations of motion, and system identification techniques. The model is represented by the transfer functions $G_\phi = \frac{\phi}{\phi_r}$, $G_\theta = \frac{\theta}{\theta_r}$, $G_\psi = \frac{\psi}{\dot{\psi}_r}$, and $G_h = \frac{h}{\dot{h}_r}$. The angles ϕ, θ, and ψ are the rotorcraft roll, pitch, and heading respectively. Rotorcraft height is represented by h. The reference commands ϕ_r, θ_r, $\dot{\psi}_r$, and $\dot{h}_r$ are for roll angle, pitch angle, heading rate, and vertical speed, respectively.

The outer-loop controllers K_u, K_v, K_ψ, and K_h in Fig. 7(a) and Fig. 7(b) are implemented to track the desired axial velocity u, lateral velocity v, heading angle ψ, and height h, respectively. Note that u and v are expressed in the local vehicle frame. Guidance and control strategies for the delivery rotorcrat follow the same approach.



(a) Heading and height control

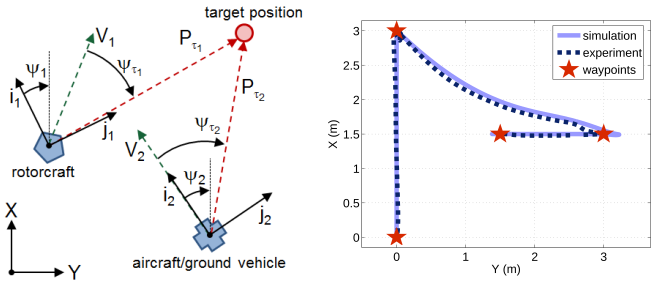
(b) Velocity control

(c) Waypoint control

Fig. 7. Elements of AR.Drone guidance and control

A point mass models the ground vehicle as a free velocity vector moving in the plane at zero height. The model inputs are the magnitude and angular rate of the vehicle velocity vector.

Although the various vehicles have different characteristics, their guidance approach is similar. The waypoints and position measurements are given in geodetic coordinates (latitude, longitude, altitude) and the objective of the guidance system for each vehicle is to minimize the distance between the current waypoint and position. For each vehicle, a navigation system transforms the geodetic coordinates to an inertial Cartesian coordinate system XYZ, whose origin coincides with the initial position of the vehicle. The XYZ coordinate system is also used for local navigation of the sensory rotorcraft for which geodetic coordinates may not be available. Fig. 8(a) shows a schematic of the positions, orientations, and velocity vectors for the various vehicles in the XY plane of the XYZ coordinate system.



(a) Coordinate frames and angles for different vehicles

(b) Trajectory tracking of AR.Drone at 1.5 m height

Fig. 8. Planar motion

The magnitude of the axial and lateral velocities u and v are used to control the direction and orientation of the vehicle velocity vector V and guide the rotorcraft to its desired position. The vehicle is driven to a desired position $[X_{ref}, Y_{ref}]^T$ using a position estimate $[X, Y]^T$(Fig. 7(c)). The position error or relative position between vehicle and target, expressed in the inertial frame XYZ, is transformed to the local frame ijk via the coordinate tranformation T^{-1}. The instantaneous relative position vector $P_\tau = [\epsilon_i, \epsilon_j]^T$, expressed in the vehicle moving frame, is multiplied by a gain K and the resulting signals are applied to the velocity references, that is, $[u_{ref}, v_{ref}]^T = KP_\tau$. Note that the commanded velocities are zero when the drone reaches the target.

For the fixed wing aircraft, the lateral velocity is negligible ($v = 0$) and the length of vector $V = u$ is set to be constant. The roll angle and resulting heading rate $\dot{\psi}$ are used to minimize ψ_τ and the distance $|P_\tau|$ between the vehicle and desired position.

For the ground vehicles the lateral velocity is also zero. The length of the axial velocity $V = u$ and heading rate $\dot{\psi}$ are used to drive the vehicle to a desired position.

4.2 Model-Based Design

The AR.drone sensory rotorcraft is a low-cost commercially available rotorcraft. Its open application programming interface (API) for vehicle control via WiFi make the drone particularly suited for rapid algorithm development.

Transfer functions G_ϕ, G_θ, G_ψ, and G_h that model drone dynamics are derived via experimental input-output data of the vehicle motion and the System Identification Toolbox™ (Ljung (2012)). The model of the vehicle dynamics is used to design the inner-loop controllers K_ϕ, K_θ, K_ψ, and K_h as well as the guidance algorithms.

Guidance relies on a rudimentary position estimate from integrating vehicle velocities expressed in the inertial frame (see Fig. 7(c)). Therefore, errors in the velocity measurements accumulate in the position estimate. An improved implementation of the system should employ direct position measurements together with a state estimator.

Guidance and control are simulated in Simulink and correctness and robustness are validated by real-time execution using the experimental vehicle. The wireless con-

nectivity allows the control used in simulation to directly control the physical AR.Drone by sending commands to a UDP network block in Simulink. This configuration enables quick design, testing, and modification of the drone control with an almost immediate turn-around of control and experiment changes while in the field. Figure 8(b) shows results of an AR.Drone tracking experiment. The measured experimental data accurately matches the simulation data, validating the mathematical models of the vehicle dynamics as well as the control and guidance laws.

Model-Based Design applies as much to the other vehicles and is the topic of an extensive body of related work (e.g., see Dorobantu et al. (2013) for fixed wing aircraft).

5. CONCLUSIONS

Exploiting technology for emergency response holds the promise of improved performance as much as equalizing disparity between nations. An automated emergency response system is documented that builds on the emerging cyber-physical systems paradigm. The system integrates autonomous vehicles of various types (ground vehicles, fixed wing aircraft, delivery rotorcraft, and sensory rotorcraft) and is structured according to: (i) mission planning, (ii) trajectory generation, (iii) guidance, and (iv) control. To plan a time optimal mission, vehicles are characterized in terms of their longevity on a mission, their speed of travel, and the payload they can carry. The vehicles are assigned locations that they should serve, after which a trajectory of waypoints is generated for each vehicle. Guidance and control allow the deployed vehicles to autonomously execute their assigned missions.

Physical vehicles or their models are connected based on internet technology to support complete integration of the physical and modeled worlds in cyber space. Coordinates and state information of each of the deployed vehicles is communicated to a Google Earth interface so operators have an integrated view of the mission as it unfolds. Experiments using the testbed, with both computational resources and experimental vehicles, demonstrate the feasibility of increased emergency response automation.

In a community sense, this article highlights the notion of a societal engineer introduced by Kaczmarczyk (2011). Using computer science and engineering provides a powerful foundation to orchestrate expertise from all disciplines and domains. If such a computational background is accompanied by the passion for addressing society's challenges, improving quality of life through innovation, and acknowledging the need for organizational missions, the results may expedite progress greatly (Zander and Mosterman (2013)).

REFERENCES

Beetz, M. (2005). Towards comprehensive computational models for plan-based control of autonomous robots. In W.S. Dieter Hutter (ed.), *Mechanizing Mathematical Reasoning: Essays in Honor of Jörg H. Siekmann on the Occasion of His 60th Birthday*, 514–527. LNCS 2605.

Careem, M., De Silva, C., De Silva, R., Raschid, L., and Weerawarana, S. (2006). Sahana: Overview of a disaster management system. In *International Conference on Information and Automation (ICIA)*, 361–366.

Cassandras, C.G., and Paschalidis, I.Ch. (2011). Optimizing the Transportation Systems Response Capabilities, *J. of Homeland Security*.

Dorobantu, A., Johnson, W., Lie, F., Taylor, B., Murch, A., Paw, Y.C., Gebre-Egziabher, D., and Balas, G. (2013). An airborne experimental test platform: From theory to flight. In *ACC, 2013*, 659–673.

Falconer, J. (2013). Honda Developing Disaster Response Robot Based on ASIMO. *IEEE Spectrum*.

Federal Emergency Management Agency (2010). Unit 4: Damage assessment. State Disaster Management Course, Emmitsburg, MD.

Hu, X. and Azarnasa, E. (2012). Progressive simulation-based design for networked real-time embedded systems. In *Real-Time Simulation Technologies: Principles, Methodologies, and Applications*, chapter 7, 181–198. CRC Press, Boca Raton, FL.

IPCC Panel (2013). The Fifth Assessment Report (AR5) of the United Nations Intergovernmental Panel on Climate Change (IPCC), Climate Change 2013: The Physical Science Basis, IPCC WGI AR5. Technical report.

Kaczmarczyk, L. (2011). *Computers and Society: Computing for Good*. Taylor & Francis, Boca Raton, FL.

Kress-Gazit, H., Fainekos, G., and Pappas, G.J. (2009). Temporal-logic-based reactive mission and motion planning. *IEEE Transactions on Robotics*, 25(6), 1370–1381.

Lee, E.A. (2008). Cyber physical systems: Design challenges. In *Intl. Symp. on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC)*.

Ljung, L. (2012). System Identification Toolbox 2012 User Guide. Technical report, The MathWorks.

MathWorks® (2013a). *MATLAB® and Simulink® product families*. March, 2013.

MathWorks® (2013b). *MATLAB® and Simulink® product families*. September, 2013.

McEntire, D.A. (2005). Disaster response operations and management: Session 20. University of North Texas, Denton, TX.

Microsoft (2008). Eagle—applying microsofts citizen safety architecture using groove and sharepoint.

Mosterman, P.J. and Zander, J. (2014). A cyber-physical system study—challenges when embedded software systems collaborate. *Software and Systems Modeling*. To appear.

ScienceDaily (2005). Science news. researcher: Global warming not to blame for tsunami. *Science News*.

Steering Committee for Foundations in Innovation for Cyber-Physical Systems (2013). Strategic Opportunities for the 21st Century. Technical report, NIST.

Waibel, M., Beetz, M., D'Andrea, R., Janssen, R., Tenorth, M., Civera, J., Elfring, J., Gálvez-López, D., Häussermann, K., Montiel, J., Perzylo, A., Schießle, B., Zweigle, O., and van de Molengraft, R. (2011). RoboEarth - A World Wide Web for Robots. *Robotics & Automation Magazine*, 18(2), 69–82.

Yin, J., Lampert, A., Cameron, M., Robinson, B., and Power, R. (2012). Using social media to enhance emergency situation awareness. *IEEE Intelligent Systems*, 27(6), 52–59.

Zander, J. and Mosterman, P.J. (2013). *Computation for Humanity: Information Technology to Advance Society*. Taylor & Francis, Boca Raton, FL.