Analysis and Design of Hybrid AI/Control Systems

Carl Glen Henshaw and **Robert M. Sanner**
Space Systems Laboratory
University of Maryland, College Park
machine@ssl.umd.edu

Abstract

This paper discusses the nature of interactions between high–level intelligent agents and low–level control algorithms. Control algorithms offer the promise of simplifying a dynamic system's apparent behavior as perceived by an intelligent agent, thus making the agent's task much easier. However, the coupled dynamics of such a hybrid system can be difficult to predict and may lead to undesirable behavior. We demonstrate that it is possible for a rational intelligent agent acting on a well–controlled dynamical system to cause undesirable behavior when coupled, and present a method for analyzing the resulting dynamics of such coupled, hybrid systems. A technique for alleviating these behaviors using newly developed control algorithms is then suggested. These controllers, which are adaptive in nature, also suggest the possibility of "distributing" learning and intelligence between the high and low levels of authority. A new architecture for hybrid systems encapsulating these ideas is then suggested.

Introduction

The deployment of dynamically complex autonomous systems such as NASA's New Millennium spacecraft series and the military's AUVs and UAVs has created the need for "high level" intelligent software agents which will interact with "low level" hardware control systems. The low level control algorithms are designed to simplify the input/output behavior of a complex dynamical system as perceived by a high–level intelligent agent, while the high level "intelligence" is responsible for specifying the desired system behavior which will satisfy top level goals and constraints. The dynamical simplification accomplished by the low level controller potentially makes the high level agent's task much easier, reducing the scope of potential behaviors which must be accommodated by goal directed plans.

Often, the lower–level software modules are treated as "black boxes" by the higher–level software; however, the dynamic behavior of a control algorithm coupled with a physical system can still be quite complex. Therefore, when intelligent agents and control algorithms are linked in a feedback system, undesirable behavior may result even though each module can be verified to function correctly in isolation. Some control algorithms are more successful than others in simplifying the apparent dynamics of the physical system; careful design of the control algorithm is therefore one key to producing desirable overall system performance.

This paper discusses both the problems incurred by unwanted AI agent/control system coupled dynamic interactions, and also the possibility for improved system performance through careful design of the coupled dynamics. Our principle interest is in automating the task of docking two space vehicles, as illustrated by the (terrestrial) research prototype in Figure 3, which is currently human piloted. We use a well–known problem in flight dynamics — pilot–induced oscillations, or PIO — as an example of the undesirable, possibly catastrophic problems that can occur through the interaction of higher– and lower–level systems. Methods for analyzing PIO–like behavior in a prototype hybrid agent/control system are presented, thus possibly providing a bridge between the AI and controls communities.

We also describe possible strategies for mitigating the effects of unwanted PIO–like behaviors. We discuss new nonlinear control technologies which can be shown to be less susceptible to these behaviors. Unlike more standard linear controllers, though, these nonlinear controllers require an accurate mathematical model of the physical system. This motivates a discussion of *adaptive* controllers which can learn system models in situ. Adaptive controllers further allow for the possibility of distributing learning and intelligence through each of the two layers of software so as to maintain desirable coupled dynamics. Finally, we suggest a new architecture for hybrid AI agents/control software, along with challenges that this architecture presents for researchers.

Hybrid AI/control systems

While it is difficult to describe a "typical" autonomous system, a sufficiently general block diagram might be similar to Figure 1. The box labeled "controlling agent" can have many instantiations, depending on design philosophy, available resources, and the exact requirements for the system's performance. A common — indeed prototype — instance of this agent might be a skilled (or unskilled) human, flying an aircraft or docking a spacecraft; the goal of the considerations in this paper is to replace this most "canonical" of agents with fully autonomous software of comparable (or superior) capability. The agent may be given a set of goals or be capable of generating goals internally, or it may be designed to instinctively perform actions which embody the goals of the designer. It may also internally generate plans

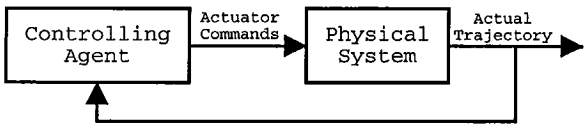

Figure 1: Block diagram of a generic autonomous system

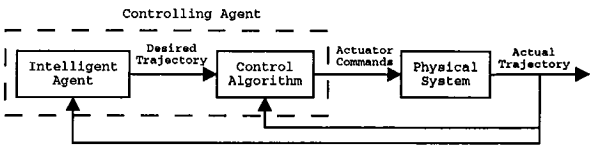

Figure 2: Block diagram of a hybrid AI/control system. The dashed box shows components encapsulated by the "controlling agent" block of Figure 1.

intended to fulfill these goals and desired trajectories to carry out the plans. Regardless of the nature of the agent or the sources of the goals which drive it, its outputs are the actuator commands intended to cause the dynamical system to follow trajectories consistent with its goals. In general, we assume that the agent receives feedback from the environment which includes information about the performance of the physical system.

In attempting to automate the process of mapping goals to physical inputs, it is helpful to break the problem into two subtasks: the problem of translating goals into sequences of desired actions, and the problem of translating desired actions into the physical inputs required to force the system to carry out these actions. This decomposition is shown schematically in 2. The output of both of these subtasks may vary continuously in response to the evolution of the actual physical state of the system, as indicated by the nested feedback loops.

While this division of labor may be artificial in terms of how the performance of the agent of Figure 1 is achieved in biological systems, it is nonetheless a useful intermediate abstraction, amenable to rigorous analysis. In fact, one could argue that the decomposition between Figure 1 and Figure 2 represents a current distinction between two classes of engineering expertise: control theory deals with the design of the second block of this diagram and analysis of the performance of the inner feedback loop, while artificial intelligence theory is often broadly directed towards design of the first block and analysis of the performance of the outer feedback loop. Moreover, as we shall discuss at the end of this paper, even if the indicated partition is non–biological, it may serve as a useful framework within which biological capabilities may be closely mimicked.

While the box labeled "control algorithm" can have many instantiations, control laws as a class are usually mathematically derived algorithms which accept as their input a desired trajectory through the physical system's state space, and output commands to the system's actuators which attempt to cause it to follow the trajectory. Control algorithms have proven successful in controlling a wide range of physical systems, often in the face of dynamic, changing environments, noisy sensors and actuators, and poorly modeled target systems. Control algorithms may be used as shown in Figure 2 to simplify the apparent dynamics of the physical system as seen by the intelligent agent; indeed, there are aircraft which are difficult or impossible for a human agent to pilot without a continuous controller which simplifies the dynamics ("handling qualities") perceived by the pilot.

Ideally, this arrangement should make the agent's task much easier in that it can rely on the control algorithm to force the physical system to carry out desired actions, even if the physical system has very complex dynamics and complicated mappings between actuator inputs and system outputs. However, even in the very simple case of a physical system which can be accurately modeled by linear differential equations and a linear proportional–integral–derivative (PID) control algorithm, the coupled control algorithm/physical system can exhibit nontrivial dynamic behavior. In particular, control algorithms require nonzero time to bring the system near a desired state or to respond to system disturbances. In addition, the manner in which the system transitions from the initial state to the desired state often entails a damped oscillatory trajectory with an overshoot that can be a significant fraction of the difference between the initial and desired states. Systems that must be modeled by nonlinear equations — the more common case, which includes such physical systems as robotic arms, spacecraft, aircraft, and underwater vehicles — can exhibit even more complex dynamic behavior (Slotine & Li 1991, pp. 4–12).

A related, but more subtle, challenge involves the unexpected behaviors that can result from the coupled dynamics of a rational intelligent agent commanding a well–controlled vehicle. On first glance, the challenge of mathematically analyzing the dynamics of a system like the one in Figure 2 may seem intractable due to the inherently non–mathematical nature of most intelligent agents. For insight on how this might be accomplished, one may take as inspiration research performed on what has been referred to as "the senior flying qualities problem": pilot–induced oscillations, or PIO (McRuer 1995, p. 2). PIO is a rapid oscillatory motion which can beset a piloted aircraft under certain circumstances, often with disastrous results. PIO is most often observed with very experienced, well–trained pilots flying a wide variety of aircraft, which have included the M2–F2, X–15, YF–22, MD–11, C–17, and the Space Shuttle (McRuer 1995, p. 2).

The PIO phenomenon is not caused by a poorly behaved dynamical system; in fact, PIOs have been observed on systems which are quite well behaved in isolation. Instead, PIO is known to result from the coupled interaction of the pilot and the underlying system dynamics (McRuer 1995, p. 3); that is, it is an instability phenomenon introduced by the closing of the outer feedback of Figure 2. Humans are often considered the "gold standard" for intelligent agents, and aircraft dynamics share many physical characteristics with the many complex systems that are now the subject of automation research. If a well–trained, highly experienced and

Figure 3: MPOD performing a docking task

supremely rational human pilot can interact with a mechanical system in such a way as to cause sudden, violent behavior, the possibility that an intelligent agent acting on an underlying dynamical system could result in undesirable behavior should at least be entertained.

Analysis of PIO in a hybrid system

The Space Systems Laboratory (SSL) at the University of Maryland, College Park has a long history of studying the operation of human–piloted spacecraft.The primary research tool of the SSL is the Neutral Buoyancy Research Facility. Neutral buoyancy, the use of water to simulate weightlessness, is the principal method used by NASA and other space agencies to practice end–to–end space operations. The SSL historically has concentrated on developing robotic systems to assist humans or, in some cases, perform space operations tasks which humans cannot or do not desire to perform. In general SSL research utilizes these robots operating in neutral buoyancy to analyze human/robotic interaction, demonstrate advanced spacecraft control algorithms, and develop autonomous robotic systems.

The specific target of the research described below is MPOD (the Multimode Proximity Operations Device), a neutral buoyancy version of an orbital maneuvering vehicle. Orbital maneuvering vehicles have been proposed for ferrying cargo and personnel between spacecraft in different orbits, for positioning astronauts at worksites during extravehicular operations, and for retrieving malfunctioning space hardware from remote locations. As such, OMVs must actively maneuver with great precision and reliability while in close proximity to other spacecraft and astronauts.

To render the apparent dynamics of this complex vehicle as simple as possible to the pilot, a number of different control strategies have been explored. Only the vehicle's attitude is currently controlled automatically, as the neutral buoyancy tank does not currently provide the sensors (GPS or equivalent) necessary to implement a translational feedback algorithm. The attitude control algorithms implemented on MPOD include a standard linear trajectory–tracking attitude controller and a family of new nonlinear trajectory–tracking attitude controllers. These latter are derived from an underlying PD component, with additional

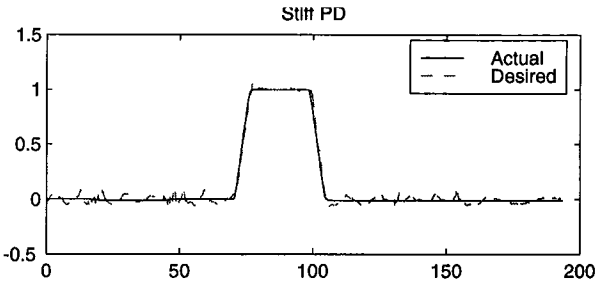

Figure 4: Example stiff PD controller step response. Dashed line is the requested trajectory; solid is the actual trajectory.

nonlinear terms in the control algorithms designed to directly compensate for the nonlinearities in the vehicle's dynamics.

The PD controllers were tuned with three different gainsets — a "loose" set, a "medium" set, and a "stiff" set — each resulting in closed–loop dynamics with similar overshoot characteristics to a step change in orientation, but with increasing bandwidth. Theoretically, these controllers would be expected to exhibit more accurate tracking and faster responses as the gainsets increase in stiffness. The loose and medium gainsets were also used for the PD components of the nonlinear controller; the action of the added nonlinear terms used sufficient additional control authority that the stiff gainset could not be employed without saturating the actuators on the vehicle.

Figure 4 shows two step responses for the stiff PD controller. Notice that the controller appears to be quite well behaved. There is no oscillatory behavior present, with an overshoot of only 7%. The system exhibits little or no steady–state offset, with the deviations from the desired trajectory being caused by noise and disturbances largely due to currents and eddies in the neutral buoyancy tank. The loose and medium gainset produced responses with similar overshoot, longer settling times, and slightly higher steady–state errors, in accordance with theory. Following the step response evaluations, the pointing accuracy of each of the controllers was evaluated on a more demanding task by commanding MPOD to track a sinusoidally varying ("tumbling") attitude maneuver. The mean–square pointing error in each control mode is shown in Figure 5. The tracking metric here is expressed in quaternion units, and may roughly be converted to degrees by multiplying by 120. Notice that the mean–square tracking error decreases as the gains increase, and that the nonlinear controllers are more accurate than the linear controllers with the same PD gainset; this again is as predicted by theory. Notice, also, that the stiff PD controller is the most accurate of any of the controllers tested.

Thus, by the generally accepted metrics used by control engineers, each of the controllers appears to result in a successful design of the inner loop of Figure 2, with the high gain PD controller producing the best overall response in the *isolated* task of tracking a specified reference attitude. Next, the *coupled* performance of a human pilot performing a docking task with each of the different controller modes

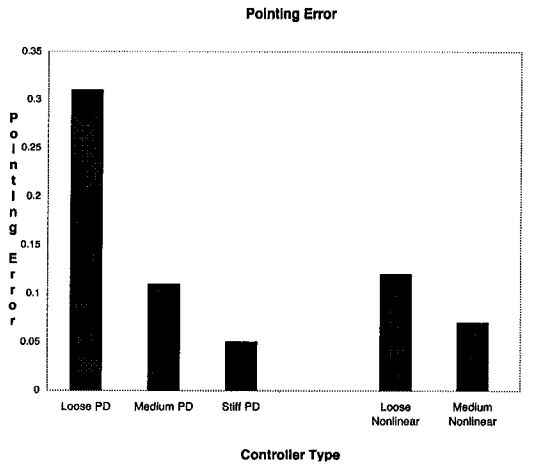

Figure 5: Pointing accuracy of linear and nonlinear controllers. Metric is quaternion error.

was investigated.

The setup for this task is illustrated in Figure 6: the pilot commands the desired motion of MPOD using two 3DOF joysticks (one for attitude, one for translation). Pilots were instructed to fly MPOD from a point in the middle of the NBRF tank to a hard dock with a target affixed to the tank wall. During testing, completion time and pilot hand controller usage (a measure of physical workload) were recorded. In addition, after using each controller the pilots were asked to rate the system performance using a Cooper–Harper questionnaire, a popular method for determining pilot mental workload (Sanders & McCormick 1993).

Although it was expected that human pilots would perform better with more accurate controllers, this proved not to be the case. Figure 7 shows a graph indicating the performance of four experienced pilots after six docking attempts with each controller. The metric in this graph factors in both mental and physical effort multiplied by task completion time as described in (Henshaw 1998, pp. 76–78). Clearly, lower values for this metric are indicative of better performance. The individual metrics also broadly tracked the trends seen in Figure 7. Notice that while performance was uniform with the nonlinear controllers, it varied quite considerably across the three PD controllers with the worst performance being exhibited with the stiff PD controller.

Somewhat counter–intuitively, then, the controller which provides the best performance in isolation appears to produce the worst performance when utilized with a goal–driven human pilot. Moreover, video observation of MPOD and its pilots when using this controller showed that MPOD could exhibited undamped oscillatory behavior, with the pilot "fighting" with the vehicle to try to stop the oscillations as shown in Figure 8. From the previous graphs, however, it is clear that this behavior was not caused by poor performance on the part of the controller. Instead, the oscillatory behavior appeared to be due to a coupling of the pilot and the closed–loop vehicle system. This is a classic symptom

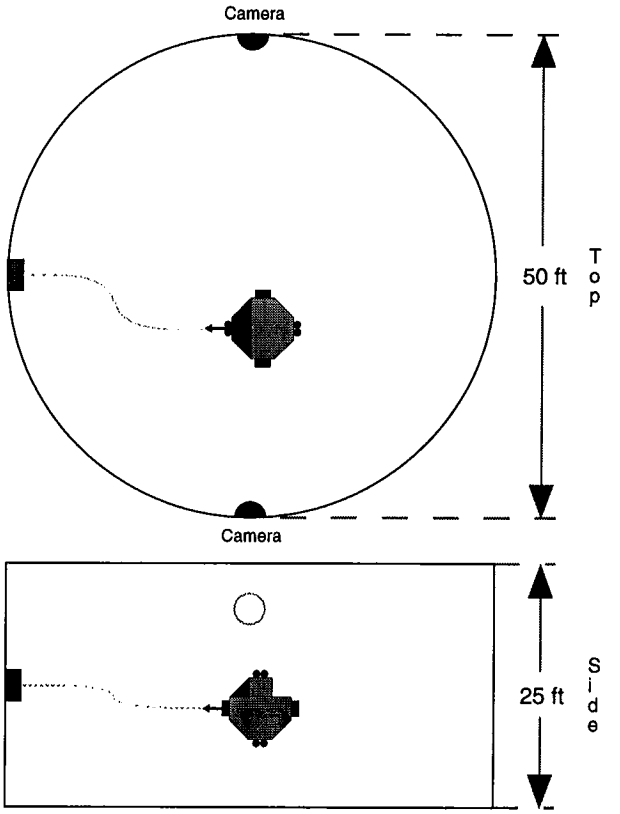

Figure 6: Schematic of NBRF tank showing the MPOD docking task (not to scale).

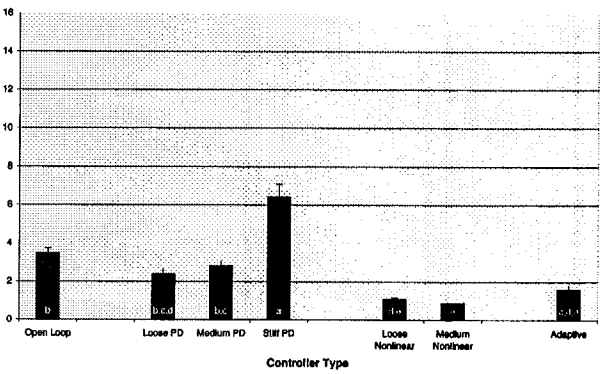

Figure 7: Pilot performance during docking tasks. Letters indicate statistically different groups.

of pilot–induced oscillations (Blakelock 1991).

In order to mathematically analyze a PIO such as that observed on MPOD, a mathematical model of human pilot behavior is needed. One widely accepted method for accomplishing this is called the "crossover model", which models the dynamics of a human pilot as a linear PD controller with a time delay. The crossover model attempts to represent the behavior of an experienced human pilot performing a set-

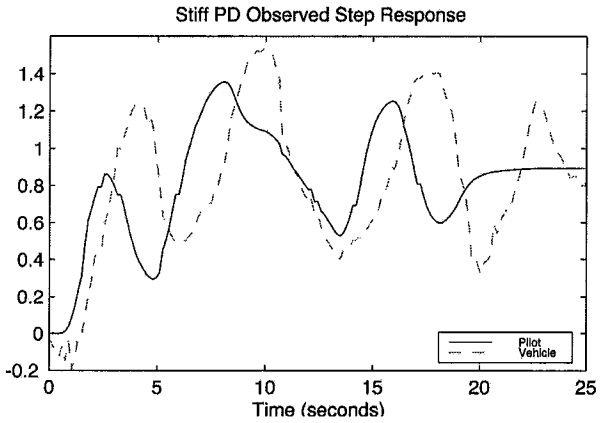

Figure 8: System behavior during a PIO event. Solid line is the requested trajectory from the pilot; dashed line is the actual trajectory of the vehicle.

point maintenance task such as maintaining straight–and–level flight in an airplane or hovering in a helicopter. The crossover model has been used within the aerospace community for at least thirty years, and has proven successful in modeling pilot behavior across a very wide range of controlled elements. For a synopsis of the crossover model and justification for its use as a human pilot model, see (Blakelock 1991) or (McRuer 1995).

Specifically, the crossover model states that the pilot's behavior (the mapping between the perceived error between the actual and goal state and the resulting commanded motion) can be modeled as

$$G_{pilot}(s) = S_p e^{-\tau_h s}(T_L s + 1) \tag{1}$$

where $G_{pilot}(s)$ is the pilot transfer function. The pilot time delay, $e^{-\tau_h s}$, represents processing delays and neuromuscular lag and is normally on the order of 0.2 to 0.4 seconds. The crossover model states that pilot's effective proportional gain S_p and derivative gain $S_p \times T_L$ change depending on the dynamics of the element being controlled so that the slope of the Bode plot of the open–loop pilot/plant system is approximately -20dB/decade in the crossover region. The gain S_p in particular is task dependent, generally increasing when the required precision to complete a task increases, as during the final phases of an attempted docking of a spacecraft or landing of an aircraft. An implied condition is that the pilot zero must be minimum phase, i.e. the pilot zero must lie in the left half–plane (Blakelock 1991, pp. 459–465).

Mathematical models of the rest of the system are also needed. From physical principles and extensive flight testing, a single–axis, linear model of MPOD was determined to be

$$G_{MPOD}(s) = \frac{2.8 \times 10^{-4}}{s\left(\frac{s}{0.35} + 1\right)} \tag{2}$$

Similarly the action of the PD controller is given by

$$\begin{aligned} \tau_{PD} &= -K_d\tilde{\omega} - K_p\tilde{\epsilon} \\ &\equiv -K_d\sigma \end{aligned}$$

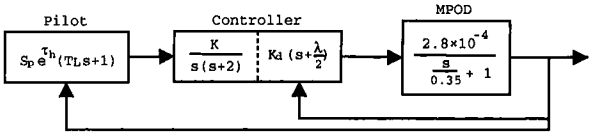

Figure 9: Block diagram of the MPOD system

where $\tilde{\epsilon}$ is a configuration error (here the attitude error) and $\tilde{\omega}$ is the corresponding velocity tracking error (here angular velocity). An alternate description of the actions of the PD controller (useful in the discussion below) employs the auxiliary variable $\sigma = \tilde{\omega} + \lambda\tilde{\epsilon}$ with complete equivalence to the first form by taking $K_P = \lambda K_D$. For the stiff PD controller, K_d is 160 and λ is 8. Finally, there is an input shaping (trajectory generation) filter between the pilot and the controller which smooths the transition between new attitudes. The filter can be represented by the transfer function

$$G_{filter}(s) = \frac{K}{s(s+2)} \tag{3}$$

where, in this instance, $K = \frac{2}{512}$.

The linearized dynamics of the controlled MPOD system as perceived by the pilot is then computed to be

$$G_{CL}(s) = \frac{K}{s(s+2)} \times \frac{\omega_n^2\left(\frac{2}{\lambda}s + 1\right)}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{4}$$

and the transfer function for the coupled open–loop dynamics of the pilot/vehicle system is

$$S_p e^{-\tau_h s}(T_L s + 1) \times G_{CL}(s). \tag{5}$$

A reprisal of Figure 2 illustrating the model just developed is shown in Figure 9.

The root locus plot is a tool that is used to examine the location of the poles (modes) of a feedback system as a function of the overall gain of the feedback loop. A linear system becomes unstable (exhibits "runaway" behavior) when any one of the poles becomes positive. The gain in the feedback system of Figure 9 is controlled by the gain of the pilot S_p, and Figure 10 shows a root locus plot for the coupled system as a function of this gain. The arrows indicate the direction of migration of the closed–loop poles as the pilot gain increases.

Notice that the system can become unstable if the pilot gain becomes large enough, as may happen in the final stages of a docking maneuver when accuracy is critical. This immediately suggests that PIO may be a problem for this system. Indeed, simulations using this model predict the oscillatory behavior experimentally observed in Figure 8 and correctly forecast the frequency of the oscillations.

This analysis illustrates that a rational "high–level" intelligent agent commanding a well–controlled system can result in unsatisfactory or even unstable system behavior. It is important to realize that the linear control methodology which produced the observed instability is virtually ubiquitous, being found an nearly all commercially available robotic systems, many flight control systems, and most

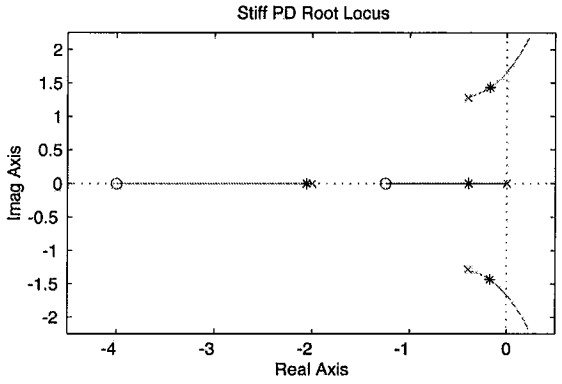

Figure 10: Root locus of coupled pilot–stiff PD controller system. Asterisks show estimated position of system closed–loop poles.

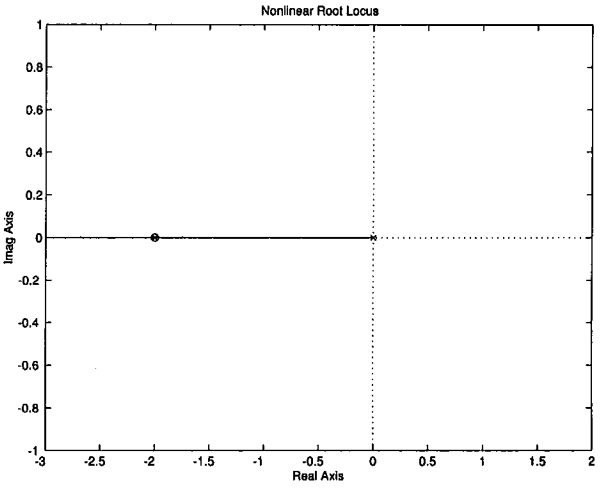

Figure 11: Root locus plot of coupled nonlinear controller/pilot system

spacecraft attitude control systems. Moreover, the dynamics of the pilot model are quite simple, and in fact might be a reasonable representation for the actions of a rational intelligent agent attempting to perform a task such as tracking a moving target.

What can be done about undesirable interactions?

In recent years there has been considerable progress in developing control algorithms which have superior performance to the linear controller analyzed above. One alternative which has also been implemented and tested on MPOD is a nonlinear controller related to the computed torque controller that is beginning to gain acceptance in the control of robotic manipulators (Slotine & Li 1991), (Egeland & Godhavn 1994), (Sanner 1996).

The form of the controller is

$$\begin{aligned} \tau_{nl} &= \tau_{PD} + H(\epsilon)\dot{\omega}_r + C(\epsilon,\omega)\omega_r + E(\epsilon,\omega) \\ \omega_r &= \omega - \sigma \end{aligned} \tag{6}$$

where σ is the auxiliary variable introduced in the above discussion of the PD controller. The additional nonlinear terms employed in this design utilize H — the positive definite symmetric inertia matrix of the dynamics, C — containing the contribution of Coriolis and centripetal forces on the dynamics, and E — representing environmental forces. Many physical systems, including aircraft, spacecraft, underwater vehicles, wheeled vehicles, and robotic manipulators, have dynamics to which this algorithm can be successfully applied. Indeed, for each of these systems, it can be (nontrivially) proven that the above control algorithm produces closed–loop dynamics (that is, the inner loop of Figure 2) which are globally stable with tracking errors which decay exponentially to zero for *any* smooth desired trajectory (Fossen 1994), (Henshaw 1998, pp. 15–30), (Egeland & Godhavn 1994).

In terms of the linear analysis above, this convergence also implies that the controller causes the closed–loop vehicle/controller dynamics to become:

$$G_{CL}(s) = \frac{K}{s(s+2)}. \tag{7}$$

Note in particular that, although the action of the controller is nonlinear, its effect is to put a particulary simple linear "face" on the closed–loop dynamics, as perceived by a higher–level agent. If the PIO analysis from above is now repeated using the nonlinear controller, the root locus plot for the outer feedback loop (representing a prediction of the coupled agent/controlled system interaction) shown in Figure 11 results. The structure of the predicted interaction dynamics predicted by this plot is quite different from that obtained with the PD controller. Disregarding the effects of pilot time delay, this system will always be stable regardless of the position of the pilot zero or the value for the pilot gain, and in fact will be critically damped for a wide range of zero locations and gains. In theory, therefore, the system should exhibit PIO only when the pilot gain becomes enormously large, in fact, an order of magnitude larger than the gains found to match the simulated responses with the observed data (Henshaw 1998, p. 107).

Notably, the nonlinear controller/vehicle system appears to be far less susceptible to pilot–induced oscillations. This does not appear to be because the controller is more accurate or has faster response times. Instead, the resistance to PIO–type behaviors appears to arise because the nonlinear controller presents a much simpler dynamical system to the agent than does the corresponding PD controller. The difference in perceived dynamic complexity can easily be seen by directly comparing equation 4, which represents the linear controller/vehicle dynamic behavior as seen by the pilot, with equation 7, the dynamic behavior of the nonlinear controller/vehicle system. It appears that systems with more complex dynamics may be more susceptible to PIO regardless of how well they perform in terms of traditional (isolated) controller metrics such as accuracy, overshoot, or bandwidth.

Desirable Hybrid System Interactions

Adaptation: low–level learning

Clearly, one challenge to using more advanced controllers such as the one described above is the problem of obtaining an accurate mathematical model of the physical system being controlled, including good estimates for the mass and rotational inertias, environmental effects, and thruster characteristics. While this can sometimes be accomplished, it may be quite challenging to completely characterize actuators, rotational inertias, or environmental forces such as drag and friction. The situation can be further complicated when the physical system changes, as with an aircraft dropping cargo or a robotic arm experiencing a partial actuator failure.

In order to alleviate this problem, adaptive versions of these controllers have been developed. The exact forms and capabilities of adaptive controllers can be quite varied depending on the exact nature of the parameters or forces being learned. One possible instantiation of such a controller is

$$\begin{aligned} \tau_{nl} &= \tau_{PD} + \hat{H}(\epsilon)\dot{\omega}_r + \hat{C}(\epsilon,\omega)\omega_r + \hat{E}(\epsilon,\omega) \\ &= \tau_{PD} + Y\hat{a} \end{aligned} \tag{8}$$

where $\hat{H}$, $\hat{C}$, and $\hat{E}$ represent approximations of H, C, and E respectively. Under the further (and not very restrictive) assumption that the estimated nonlinear terms can be factored as

$$\hat{H}(\epsilon)\dot{\omega}_r + \hat{C}(\epsilon,\omega)\omega_r + \hat{E}(\epsilon,\omega) = Y\hat{a},$$

where Y is a matrix of known nonlinear functions and $\hat{a}$ is a vector of unknown parameters, then the "adaptation law"

$$\dot{\hat{a}} = -\Gamma Y^T \sigma \tag{9}$$

guarantees global stability and asymptotically converging tracking errors, even if the "true" physical parameters a are completely unknown when the system is initialized (Egeland & Godhavn 1994). The positive definite matrix Γ is the learning rate for the system.

System parametric uncertainties such as mass, rotational inertias and drag coefficients can be handled this way. More complex structural uncertainties such as the exact nature of drag or frictional forces can be dealt with by neural–network based adaptive components, as can complex actuator and structural dynamics. Global asymptotic stability can also be shown for neural–network based adaptive controllers and for combinations of different kinds of adaptation mechanisms (Sanner 1996).

Cooperative learning in hybrid systems

Many AI designs for robotic learning model the learning process at the top level (the first box in Figure 2), assuming "dumb" low–level servo controllers which mindless push the system to the configurations requested by the (adaptive) high level agent. The above discussion shows that control theory has its own notions of learning, which enable "smart" servos that can labor to maintain the fiction of dynamic simplicity for a top level agent, despite uncertainty in the physical properties of the systems they control.

This suggests that learning can be profitably *distributed* between the two modules of Figure 2, with the low–level controller becoming incrementally better at its goal of tracking specified system configurations in a manner relatively transparent to the top level agent, while simultaneously this agent may also be incrementally learning to specify motions which better achieve its top level goals. In fact, the analysis above suggests that such a strategy might be necessary for the success of the hybrid system, as PIO–type instabilities may set in if the low–level controller does not successfully maintain the appearance of dynamic simplicity for the high level agent.

While rigorous mathematical analysis of this cooperation, in the spirit of the analysis above, will be quite complex, it is encouraging to note from Figure 7 that when the adaptive nonlinear controller was paired with a human pilot (who is presumably learning to better perform the docking task at each iteration), the overall performance metric was among the best seen in these experiments. More extensive data must be collected quantifying the interaction of a learning agent (like a human) with a learning controller like that described above to fully support the advantages of this cooperation. This is the subject of current experimentation in the Space Systems Lab.

Biological Considerations

Since human and other biological agents are, in a sense, the current "gold standard" against which potential autonomous software architectures are evaluated, it is interesting to inquire whether there is any biological support for the hierarchical interaction and distributed learning paradigm suggested above. While the data is limited, there do appear to be indications that the actions of biological intelligences might at least be profitably *modeled* in this fashion.

It has been hypothesized that humans may generate "desired trajectories" to perform certain tasks which are invariant regardless of the muscles or joint actions required to carry them out. For instance, it has been noted that a person's handwriting remains nearly the same regardless of whether the writing takes place on a piece of paper or a chalkboard, indicating a characteristic pattern of hand motions used to describe each letter. However, writing on paper requires a very different set of actuation of arm and hand muscles than does writing on chalkboard, and a different set of environmental forces is encountered (Winter 1990). While one could attempt to explain this behavior in terms of the "black box" agent seen in Figure 1, it seems more easily interpreted in the model of Figure 2: the brain may have characteristic trajectories for the hand in its goal to trace out each letter (which are subgoals of spelling a word, which are subgoals of sentences, etc.), while a low–level, adaptive control strategy works to execute these motions.

More concretely, consider a planar reaching task — that is, when the goal presented is to move the hand to a new position on a horizontal surface. In this situation, humans appear to adopt an invariant strategy of moving the hand in a straight line between the new and old positions, with a velocity profile that assumes a characteristic bell shape. This

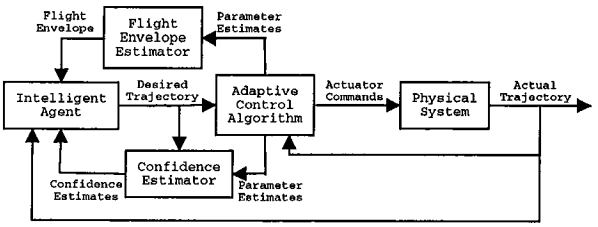

Figure 12: Suggested architecture for hybrid autonomous agent/control systems

is the so–called "minimum jerk" trajectory, in that it corresponds to a hand trajectory which minimizes the derivative of acceleration (jerk) as the hand transitions between the two endpoints (Flash & Hogan 1985). Additionally, if a deterministic external force is applied to the hand as it moves in the plane, disturbing this trajectory, with practice human subjects learn to account for the presence of this force until they can again execute the minimum jerk trajectory, despite the disturbance (Shadmehr & Mussa-Ivaldi 1994). Interestingly, rejection of the perturbation is not accomplished by "stiffening" control of the arm (just as stiffer gains in a PD controller will produce lower tracking errors), but rather by learning a model of the forces which will be applied to the arm at each point in the workspace. This was shown by suddenly removing the presence of the disturbances in subjects who had already learned to accommodate them, and observing that the resulting arm motion was perturbed away from the minimum jerk trajectory just as though the "negative" of the original disturbance was applied (Shadmehr & Mussa-Ivaldi 1994).

One possibility for modeling these observations is again in the framework of Figure 2, with the top level agent computing a minimum jerk trajectory between current and goal states, and the low level agent acting as an adaptive nonlinear controller in the fashion described in the previous section. Indeed (Kosha & Sanner 1999) performed a simulation of the perturbed reaching task in this spirit, additionally modeling the adaptive nonlinear component of the arm control dynamics using neuron–like elements, and produced results which qualitatively matched many of the reported features of the human responses.

These observations are not meant to suggest that biological signal processing is segregated into the subtasks and components discussed herein, but rather that this framework may be useful for producing systems which can mimic the important observed features of biological systems. Moreover, as demonstrated above, this framework appears quite amenable to theoretical analysis which can both predict the dangers of PIO–like behavior and provide the designer with insights which may lead to the creation of more robust adaptive autonomous systems.

Directions for Further Work

Control systems do not have unlimited "command authority": there are practical limits to the magnitude of the forces, torques, voltages, etc. which can be manipulated in the physical system to influence its actions. This means that the low–level controller can maintain the desired apparent dynamic simplicity to the high–level agent only over a finite range of configurations; in aerospace this range is sometimes known as the "flight envelope". Accurate knowledge of this envelope is an important requirement for the high–level agent, lest, in pursuit of its goals, it commands motion of which the system is incapable.

The flight envelope inherently depends on the exact dynamics of the system — the mass, rotational inertias, external forces, and so on, as well as the characteristics of the actuators. To compute the envelope of feasible trajectories, then, the high level agent must have knowledge of, or a means to learn about, the same physical parameters that an adaptive low level controller requires. Extending the cooperative learning notion suggested above, might the two modules actually share this information, for example by the adaptive controller feeding back its parameter estimates to the high level agent?

This appears to be an attractive idea at first glance, but it is complicated by a well–known feature of adaptive control algorithms: although they can guarantee accurate tracking, they *cannot* guarantee that their parameter estimates will converge to the true physical values without additional conditions. A simple example to illustrate this point might be an attitude controller attempting to estimate rotational inertias. If the vehicle is only asked to rotate around one axis, the adaptive controller does not need accurate estimates of the rotational inertias of the other axes in order to guarantee good tracking, and in fact there is no way for the adaptive mechanism to accurately estimate a rotational inertia without experiencing rotations around the corresponding axis.

One could think of this as the need for the controller to continually "practice" throughout its envelope in order to guarantee that all of its parameter estimates are exact. Mathematically, convergence of the estimated parameters can be guaranteed if the desired trajectories given to the controller are sufficiently complex to allow the controller to "explore" enough of the state space. This requirement is known as the persistency of excitation criterion (Slotine & Li 1991, pp. 331–332), and can be exactly quantified for a given physical system.

Unfortunately, it may be that the need to fulfill the persistency of excitation requirement might conflict with the system's goals. For instance, exploring large parts of the state space might involve large motions, fast velocities, or complex "wiggles" which, by themselves, do not aid in accomplishing the system's objectives and could actually violate safety constraints during certain operations like docking a spacecraft or landing an airplane. Conversely, though, generating an accurate flight envelope is necessary to accomplish future goals. The tension between optimally accomplishing near-term goals and fulfilling the persistency of excitation requirement entails a high level of cooperation between the agent and the controller. One possible approach to solving this problem might be for the agent to take "time out" between goals — for instance, during transit between worksites — to specifically command trajectories which sat-

isfy the persistency of excitation criterion. Exactly how this could be done is presently an unsolved problem, but has an intriguing parallel with children practicing in order to learn the abilities and parameters of their physical systems — their bodies.

Similarly, it may be useful for the agent to have some knowledge of the accuracy of the flight envelope estimate. If an upcoming task required rotation around a specific axis, for instance, the agent might use an estimate of the accuracy of the corresponding rotational inertia to decide whether to "practice" rotational maneuvers around that axis in order to ensure an accurate estimate before attempting the task. Because the persistency of excitation criterion is a mathematical requirement on the desired trajectories, it should be possible to use it to produce confidence intervals on the accuracy of the physical parameter estimates. Methods for generating confidence intervals, and more importantly for their use by an agent, are also matters of current research.

The considerations discussed here suggest an architecture for hybrid systems as shown in Figure 12. This new architecture allows for distributed adaptation, with the controller both simplifying the apparent system dynamics for the intelligent agent and generating estimates of the system's physical parameters. The agent uses these estimates to produce desired trajectories which fall within the flight envelope and, in addition, are sufficiently complex to allow the controller's adaptation mechanism to accurately update it's parameter estimates. Much work remains to make the ideas encapsulated by this architecture rigorous, and especially to analyze the implications of the additional feedback loops.

Conclusions

We have demonstrated that undesirable behavior can result from the interaction of rational intelligent agents and well–designed control algorithms. Mathematical methods of analysis can predict this behavior, however. These methods also lead to new design goals for the controllers in these systems, favoring controllers which provide simpler closed–loop system dynamics. We have also presented advanced controller designs which, although more complex than standard linear controllers, are more successful at simplifying the closed–loop dynamics. This appears to make them less susceptible to PIO.

We discussed adaptive versions of these controllers which eliminate the need for the designer to accurately model the dynamics of the physical system. Furthermore, these controllers naturally lead to systems with distributed adaptation which may allow for the improved performance of intelligent agents in the face of changing system parameters or changing environmental factors. Preliminary but intriguing evidence indicates that biological systems may work in a similar manner.

As more complex autonomous systems are deployed, intelligent agents will be faced with tasks which will require human–level control abilities. This work is hopefully a small step in that direction.

References

Blakelock, J. H. 1991. *Automatic Control of Spacecraft and Missiles.* John Wiley & Sons.

Egeland, O., and Godhavn, J.-M. 1994. Passivity-based adaptive attitude control of a rigid spacecraft. *IEEE Transactions on Automatic Control* 39(4).

Flash, T., and Hogan, N. 1985. The coordination of arm movements: an experimentally confirmed mathematical model. *The Journal of Neuroscience* 5.

Fossen, T. I. 1994. *Guidance and Control of Ocean Vehicles.* John Wiley & Sons.

Henshaw, C. G. 1998. Nonlinear and adaptive control strategies for improving the performance of teleoperated spacecraft and underwater vehicles. Master's thesis, University of Maryland, College Park.

Kosha, M., and Sanner, R. M. 1999. A mathematical model of the adaptive control of human arm motions. *Biological Cybernetics* 80.

McRuer, D. T. 1995. Pilot-induced oscillations and human dynamic behavior. Contractor Report N96-24466, National Aeronautics and Space Agency.

Sanders, M. S., and McCormick, E. J. 1993. *Human Factors in Engineering and Design.* McGraw-Hill.

Sanner, R. M. 1996. Adaptive attitude control using fixed and dynamically structured neural networks. In *Proceedings of the 1996 AIAA Guidance, Dynamics and Control Conference.*

Shadmehr, R., and Mussa-Ivaldi, F. A. 1994. Adaptive representation of dynamics during learning of a motor task. *The Journal of Neuroscience* 14(5).

Slotine, J.-J. E., and Li, W. 1991. *Applied Nonlinear Control.* Prentice Hall.

Winter, D. A. 1990. *Biomechanics and Motor Control of Human Movement.* Wiley Interscience Publication.