Service-Oriented Multidisciplinary Optimization of a Closed Loop Flapping Wing Subjected to Wind Gusts

Gary Clayman[1]
Data Science Automation, McMurray, PA, 15317

and

Philip Beran[2], Manav Bhatia[3], Bret Stanford[4]
U.S. Air Force Research Laboratory, Wright-Patterson AFB, OH, 45433

The optimization of a controlled flapping wing micro aerial vehicle for energy-efficient flight in gust using a general, service-oriented framework is investigated. Kinematic (wing-stroke pattern), geometric (wing shape), frequency, and control (state penalty) design variables are considered in a sequence of optimization problems. The service-oriented framework is applied to the integration of a flapping vehicle physics-based model, a linear quadratic regulator control system, a continuous gust model, a gradient based optimizer utilizing the method of moving asymptotes, and a graphical user interface to facilitate design studies. Constraints are applied to the path displacement of the vehicle and the peak control power exerted to maintain a fixed position during hover. Five optimization studies utilizing varying design parameters and gust disturbances are presented. In the optimization studies, the tradeoff between prescribing periodic kinematic motion and governing the kinematic motion with close loop control is evaluated.

I. Introduction

Throughout the past several decades flapping wing (or ornithopter) technology has accelerated due to an increasing interest in transitioning flight characteristics inherent in flying insects and birds to the Micro Aerial Vehicle (MAV). It is envisioned that by adopting these characteristics, the Flapping Wing MAV (FWMAV) will be capable of performing flight profiles considered impassible by a MAV equipped with current fixed and rotary wing technology. These profiles may require advanced agility, persistent hovering, and perching in highly constrictive environments. This includes environments such as building interiors, tunnels, and urban areas composed of narrow alleys with obstructions such as utility poles and wires, etc… Current research and development is attempting to understand and exploit flapping wing technology to meet these flight profile capabilities. In turn, this development will introduce a new generation of FWMAVs which can better perform a myriad of functions including search and rescue operations and intelligence, surveillance and reconnaissance gathering.

Flapping wing technology continues to prosper due to the coalescing results of research and development conducted across various scientific disciplines. This composite research has fomented insight into different aspects of the FWMAV. Like the fixed or rotary wing MAV, the tightly coupled multidisciplinary system of the FWMAV includes fluid dynamics, control theory, and structures; however the coupling between these areas and the aerodynamics of the flapping wing is nebulous. Researchers are still trying to fully understand the wing-to-wing and wing-to-body fluid wake interaction, their resultant forces and how to artificially apply these to obtain the desired flight dynamics. Equally as import is the power management for the FWMAV, due to their size, weight and the limitations imposed by the low power densities available in current micro power, the MAV will have at best,

[1] Consultant Research Engineer, Data Science Automation: gjc@dsautomation.com, Member AIAA
[2] Principal Research Aerospace Engineer: philip.beran@wpafb.af.mil, Associate Fellow AIAA
[3] Research Engineer, Universal Technology Corporation: manav.bhatia@wpafb.af.mil, Member AIAA
[4] Research Engineer, Universal Technology Corporation: bret.stanford@wpafb.af.mil, Member AIAA

modest available power. A number of contributions have been made to provide more understanding and solutions to these problems; Oppenheimer [8] presented such a solution in the form of a split-cycle wing beat forcing function which allow for six-degrees-of-freedom flight control using only two flapping wing actuators. Additionally, FWMAV progress has effectively leveraged from the biological science community which has contributed tremendous insight into the flight kinematics and navigational behaviors of flying insects and birds such as the Thrush Nightingale *(Luscinia Luscinia)* and Teal (*Anas Crecca)* wind-tunnel observations provided by Pennycuick [9] or the observations made by Rutkowski [11] on the navigation techniques used by the Hawk-Moth (*Manduca sexta)* in tracking pheromone plumes.

The research discussed in this paper investigates methods for adding additional insight into the question as to whether the flapping wing may provide advantages over the rotary or other applicable MAV technology. Using multidisciplinary optimization (MDO) methods this research presents kinematic and wing-shape optimization for a closed-loop flapping wing model subjected to wind gusts. This work integrates a closed-loop linear quadratic regulator (LQR) designed controller with a rigid flapping wing model developed by Bhatia, Patil, and Woolsey [3] along with Stanford and Beran [3][16] with a gradient based optimizer to minimize the power required to hover in a windy environment. In their research, Bhatia et.al developed a method for stabilizing a FWMAV operating in a lateral or longitudinal gust. This research is unique in that it presents optimization studies of the flapping wing subjected to multiple wind gust vectors under spherical displacement and peak control power constraints while varying the wing geometry, control gain coefficients, and kinematics. To encourage broader development collaboration and the utilization of remote computing resources, the optimizations are carried out over a distributed computing environment.

The models and optimization algorithm discussed in this paper were initially developed and tested using Matlab® software and then compiled as stand-alone executable modules using Matlab Compiler™. The modules were later integrated in the service-oriented framework, "Service-ORiented Computing EnviRonment" (SORCER); a federated service-to-service meta-computing environment which employs exertion-oriented programming [14]. SORCER allows a myriad of engineering applications, or in our case stand-alone executable modules, to be published as service providers where they can be called upon as part of a federated service object-oriented architecture. To facilitate this research, a previously developed "Multidisciplinary, Multifidelity, Model Based Computational Tool" (M^3CT) was utilized as a graphical interface for generating service requests to the distributed computing environment. The M^3CT serves as a tool in which to initialize, control, and monitor the progress of a study composed of multiple distributed physics based models, optimizers, and other engineering design and research applications.

As previously mentioned, the purpose of this research is to investigate methods for performing analysis related to the FWMAV; the optimization results presented in this paper have not been validated against physical models nor have the results been compared with optimization results utilizing more advanced aerodynamic computational methods. The aerodynamic models used in this study utilize a lower fidelity quasi-steady blade element method so that the aerodynamic terms can be projected onto the rigid dynamic states. Subsequently, the models do not include aeroelastic terms and therefore the optimization results may not represent a real-world wing geometry and kinematic solution due to the disjoined stress and aerodynamic effects related to changes in wing thickness, planform, and flexibility. It is also important to note that both the optimization method and its implementation approach are not exclusive to the flapping wing or the aerodynamic model used. Aside from the other reasons for studying the FWMAV previously mentioned, the FWMAV provides an advanced example of an MDO application due to the complexity of its aerodynamics, control system, structure, etc... It is the opinion of the author that this optimization method may be applied to, and would greatly benefit, other areas of aerospace research.

II. Methodology

A. Coordinate System Definition

The rigid wing and body movement, shown in Figure 1 is comprised of a fixed reference frame in the global coordinate system and the inertial frames of reference denoted by $B=\{B_X,B_Y,B_Z\}$, $I=\{I_X,I_Y,I_Z\}$, and $W_n=\{W_{nX},W_{nY},W_{nZ}\}$. The vehicle body frame B is obtained by rotating I with respect to the global frame. The body frame component $+B_Y$ extends in what is considered the vehicles normal forward flight while $+B_X$ extends out the right wing. The wing frame, denoted W_n is rotated about B to obtain the flapping stroke plane, wing pitch (about W_{nX}), and sweep as represented by the three Euler angles, ϕ, η, and θ respectively.

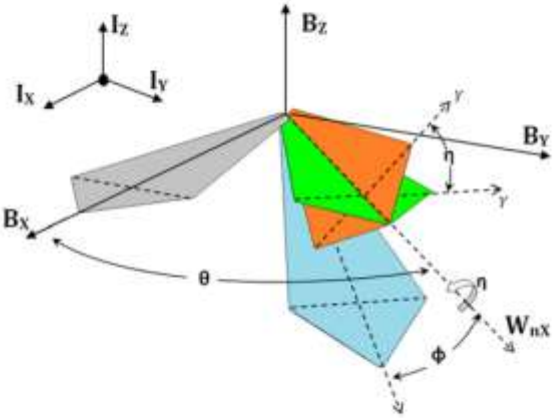


Figure 1. Flapping wing coordinate system

B. Vehicle Geometry Definition

A diagram of the rigid wing geometry is provided in Figure 2, where the wing shape derived in the work from Bhatia [3] is described using eight parameters: three chord section lengths, their respective chord thickness, span-break ratio (SBR), and the wing radius. The three chord sections lengths at the wing root, span-break position, and wing tip (denoted C_0, C_S, and C_N respectively) are orthogonal and reflected symmetrically across the center wing-span line which extends collinear with the local x-axis (W_{nX}). A wing cross section thickness is also assigned to each respective chord position; subsequently the thickness and chord lengths between prescribed chords can be determined from simple linear extrapolation. The span break ratio represents the position of the span-break chord with respect to the wing radius as measured from the wing root along the center wing-span line.

A simple symmetrical ellipsoid described by elliptic axis major and minor diameters represents the vehicle body, where the major axis defines the length of the ellipsoid and the minor represents the width of the body. Body mass is defined as is independent of the volume; however the body mass is accounted for in the inertial terms and contributes to establishing lift requirements for hover. The position of the right flapping wing hinge is defined with respect to the body center of gravity and is independent of the body geometry; subsequently the left wing is positioned symmetrically based on the right wing position. Aerodynamic effects related to the body shape are neglected as well as collision detection between the body and the flapping wing. For this research, this is easily avoided by prescribing appropriate boundaries to the rigid wing kinematics. If an aero-elastic model were introduced this would certainly be of concern and it may be prudent to implement a kinematic boundary based on wing-body collision detections.

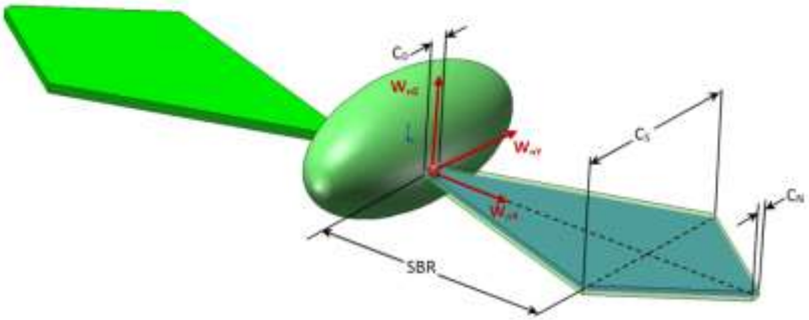


Figure 2. Flapping wing geometry description

C. Kinematic Parameterization

The flapping wing model presented in this research utilizes a kinematic parameterization that facilitates the split-cycle control approach of Doman and Oppenheimer [5]. The parameterization has a biological basis, as proposed by Berman and Wang [2] and used by Stanford et al. [15] and was extended for smooth transition between flapping cycles by Bhatia et al. [3]. The three degree of freedom flapping motion is assumed to be given by the following Euler angles (see Figure 1. Flapping wing coordinate system Figure 1) representing the flapping stroke plane (ϕ), wing sweep (θ), and pitch (η) as parameterized functions of periodic time:

$$\phi(t) = \phi_m\left(1 + A_\phi\right)\frac{\sin^{-1}\left(K_\phi \cos\beta\right)}{\sin^{-1} K_\phi} + \phi_o \tag{1}$$

$$\theta(t) = \left(1 + A_\theta \cos\left((\omega - \delta)t\right)\right)\left(\theta_m \cos(N_\theta \beta + \theta_S) + \theta_o\right) \tag{2}$$

$$\eta(t) = \lambda_m \frac{\tanh\left(K_\eta \sin(\beta + \eta_S)\right)}{\tanh K_\eta} + \lambda_o \tag{3}$$

The wing sweep frequency (equation 2) deviates from the flapping frequency (ω) by a factor of N_θ. This deviation has a strong influence on the overall path of the wingtip; for example, when $N_\theta = 1$, the wing tip follows an elliptic path, when $N_\theta = 2$, the wingtip follows a "figure-eight" path. The subscripts associated with each angle correspond to the angle obtained from the previous flapping cycle (*old)* or the angle magnitudes (*m*), phase shifts (*s*), and offset (*o*). The angle offset, is defined as the angle between the wing neutral line and the center of the angular arc length (magnitude) as shown with respect to ϕ in Figure 3; here the wing neutral line lies collinearly with the FWMAV body y-axis B_Y.

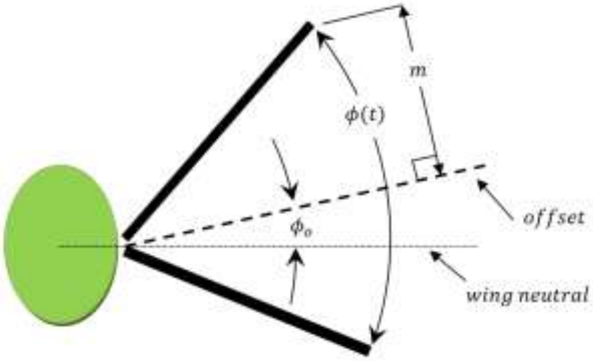


Figure 3. Wing kinematic angle magnitude and offset description

The coefficients K_ϕ and K_η define their respective Euler angle waveform function between sinusoidal ($K = 0$) and a rectangular function ($K_\phi = 1$) for the flapping stroke plane or a triangular function ($K_\eta = 1$) for the wing sweep or pitch. The term δ represents the split-cycle control parameter, which adjusts the frequency between the up and down stroke while maintaining a constant period of $2\pi/\omega$. The time dependent coefficients $\hat{\eta}_m$, $\hat{\eta}_o$, β, and the amplitude scaling factors A_ϕ and A_θ are provided to ensure continuity of flapping motion across consecutive wing-beat cycles.

$$\hat{\eta}_m = \begin{cases} \eta_m \dfrac{(\omega - \delta)2t}{\pi} + \eta_{m,old}\left(1 - \dfrac{(\omega - \delta)2t}{\pi}\right) & 0 \le t < \dfrac{\pi}{2(\omega - \delta)} \\ \eta_m & \dfrac{\pi}{2(\omega - \delta)} \le t < \dfrac{2\pi}{\omega} \end{cases} \tag{4}$$

$$\hat{\eta}_o = \begin{cases} \eta_o \dfrac{(\omega - \delta)2t}{\pi} + \eta_{o,old}\left(1 - \dfrac{(\omega - \delta)2t}{\pi}\right) & 0 \le t < \dfrac{\pi}{2(\omega - \delta)} \\ \eta_o & \dfrac{\pi}{2(\omega - \delta)} \le t < \dfrac{2\pi}{\omega} \end{cases} \tag{5}$$

$$\beta = \begin{cases} (\omega - \delta)t & 0 \le t < \dfrac{\pi}{\omega - \delta} \\[2ex] \dfrac{\omega(\omega - \delta)t - 2\pi\delta}{\omega - 2\delta} & \dfrac{\pi}{\omega - \delta} \le t < \dfrac{2\pi}{\omega} \end{cases} \tag{6}$$

$$A_\phi = \begin{cases} \dfrac{\phi_{m,old} + \phi_{o,old} - \phi_o}{\phi_m} - 1 & 0 \le t < \dfrac{\pi}{2(\omega - \delta)} \\[2ex] 0 & \dfrac{\pi}{2(\omega - \delta)} \le t < \dfrac{2\pi}{\omega} \end{cases} \tag{7}$$

$$A_\theta = \begin{cases} \dfrac{\theta_{m,old} \cos\theta_s + \theta_{o,old}}{\theta_m \cos\theta_s + \theta_o} - 1 & 0 \le t < \dfrac{\pi}{2(\omega - \delta)} \\[2ex] 0 & \dfrac{\pi}{2(\omega - \delta)} \le t < \dfrac{2\pi}{\omega} \end{cases} \tag{8}$$

The kinematics shown represents only a single wing; a similar set of parameters describes the motion of the second wing. The resultant prescribed kinematic parameters that fully define the time varying motion of the flapping wings are:

$$X_{kin} = \{\ \phi_m,\ \phi_o,\ K_\phi,\ \theta_o,\ \theta_m,\ \theta_S,\ \eta_o,\ \eta_m,\ \eta_S,\ K_\eta\ ,\ \omega,\ \delta\ \} \tag{9}$$

As presented by Bhatia et al [3], the time dependent scaling factors are applied at the beginning quarter of the cycle to ensure that the wing-beat offset values are reflected in the current cycle rather than the following cycle, as is the case in the original split-cycle method by Doman and Oppenheimer [5]. Additionally, the scaling factor Bhatia introduced incorporates the amplitude parameters ϕ_m and η_m for the wing flapping stroke plane and pitch respectively, as opposed to the originally fixed values of π/2 and π/4. The flapping stroke plane over a single wing-beat is shown in Figure 4, where the modified Bhatia (et al) [3] implementation is compared with the original Doman and Oppenheimer split cycle waveforms [5] for values of δ less than zero (left) , equal to zero (center), and greater than zero (right). The first quarter cycle scaling factor plot depicts the transition from a flapping stroke magnitude of π/3 found in the previous cycle to the current flapping stroke magnitude of π/2. The remaining waveform, which includes the impeded upstroke and advanced down stroke, remains consistent between the two approaches. For this research, the split-cycle term δ is held at zero throughout the optimizations.

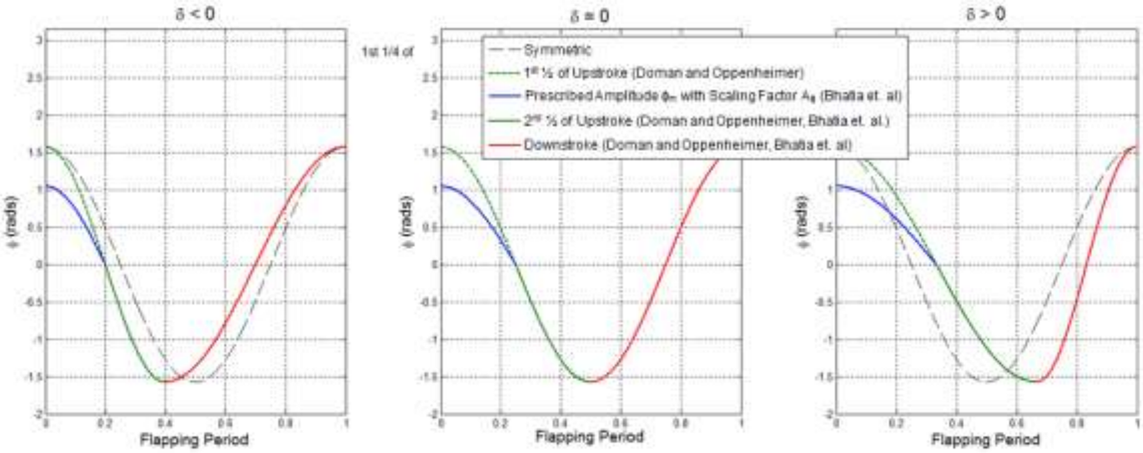


Figure 4. Split-cycle waveform (Doman and Oppenheimer [5]) with scaling factor and prescribed amplitude (Bhatia et. al [3])

D. Control System Modeling

For obtaining closed-loop control for stabilized orbital position in the presence of gust disturbances, the flapping wing model relies on a linear quadratic regulator (LQR) for calculation of an optimal controller gain matrix (***K***) such that the feedback changes to the wing kinematics (***q***) that relate to the current state vector (***x***) are given by:

$$d\boldsymbol{q}^{t_0} = -\boldsymbol{K} d\boldsymbol{x}^{t_0} \qquad (9)$$

This is accomplished first by modeling the FWMAV as a non-linear periodic system using Floquet analysis to find the periodic trim. The periodic shooting method is then applied to obtain the trimmed periodic orbit. This orbit is then converted to a discrete linear representation of the system about the trim orbit which also coincides with the period of the flapping cycle. As a result, the state of the FWMAV can be defined by its position (both linear and angular) and respective time derivatives related to both the inertial frame and body frame as shown in the position vector:

$$\boldsymbol{x}^{t_0} = \{x_I \quad y_I \quad z_I \quad \theta_x \quad \theta_y \quad \theta_z \quad \dot{x}_B \quad \dot{y}_B \quad \dot{z}_B \quad \dot{\theta}_{x,B} \quad \dot{\theta}_{y,B} \quad \dot{\theta}_{z,B}\} \qquad (10)$$

As shown by Bhatia et. al [3], given the position vector ***x*** (above) and the kinematic vector ***q***, the sensitivity of the states at the end of each flapping cycle with respect to the state at the beginning of the flapping cycle can be shown as a linearized discrete-time system of equations in the form of:

$$d\boldsymbol{x}^{T+t_0} = d\boldsymbol{x}^{t_0}\left[\frac{dx}{dx^{t_0}}\right]_{t=T+t_0} + \; d\boldsymbol{q}^{t_0}\left[\frac{dx}{dq^{t_0}}\right]_{t=T+t_0} + d\boldsymbol{q}^{t_0-T}\left[\frac{dx}{dq^{t_0-T}}\right]_{t=T+t_0} + \Delta^{t_0}\left[\frac{dx}{\Delta^{t_0}}\right]_{t=T+t_0} \qquad (10)$$

For which the kinematic vector $d\boldsymbol{q}^{t_0-T}$ contains only the kinematic parameters from the previous cycle that influence the state variables from the current cycle. The kinematic parameters which may be used for trim are prescribed in the model setup. To establish an effective gain matrix, LQR theory attempts to identify those values for ***K*** which best minimizes the linear quadratic cost so that:

$$\boldsymbol{K} = \rho\boldsymbol{R} + \; \boldsymbol{B}^T\boldsymbol{P}\boldsymbol{B}^{-1}\boldsymbol{B}^T\boldsymbol{P}\boldsymbol{A} \qquad (11)$$

In the equation above, ***P*** is obtained from the solution to the *discrete algebraic Riccati* equation, while the matrices ***A*** and ***B*** are the system matrix and control coefficient matrix (respectively) of the discrete linear time-invariant system model. The coefficient ρ is of particular importance in this research, because it represents requital for control-cost in the form of state-penalty. Subsequently, as ρ increases the emphasis on minimizing control cost increases; conversely, as ρ decreases, the emphasis on minimizing state cost increases. More discussion on ρ and its impact related to optimization is discussed later. Furthermore it is from the state sensitivities described above that we also calculate the cycle average power related to the baseline kinematics along with the power sensitivities related to the control inputs, which are discussed in further detail later as they relate to the optimization.

E. Wind Gust Disturbance Modeling

To generate disturbances in the orbital flight of the FWMAV, this research utilizes the $(1 - Cos)$ linearized discrete gust model with gust vectors distributed over a spherical boundary enclosing the FWMAV. Each gust is defined by three parameters: its unit vector, magnitude, and frequency. The gust disturbance can be analyzed as the result of a single vector or the statistical average over multiple gust vectors applied laterally, longitudinally, or spherically. For a linearized gust model, the FWMAV kinematic perturbations are linearly proportional to the gust magnitudes; since the response is the product of the forcing function. The resultant disturbance may be scaled according to the gust magnitude, eliminating the need for multiple gust velocity optimizations. To ensure complete capture of the entire gust cycle, the number of flapping cycles is calculated as a function of the gust frequency (f_G) as shown in equation 12.

$$N_{Flapping\ Cycles} = \frac{\omega}{2\pi f_G} \qquad (12)$$

From equation 12, it can be shown that lower gust frequencies will have a more persistent effect on a FWMAV displacement for an equal number of flapping cycles. In cases involving multiple linear gust vectors, nodes are equally distributed laterally or longitudinally over a semi-circle as shown in the left of Figure 5 or over a semi-sphere as shown in the right of Figure 5. Equally distributing spherical coordinates can lead to tighter clusters of distributed vectors at the antipodal points representing the axis of rotation. To avoid this, the equal spherical distribution of nodes is calculated using the golden ratio; ensuring that the distance between all adjacent nodes are the same. The resultant gust vectors are located at the node points (as shown by the arrows in the left of Figure 5) directed radially towards the center of the spherical orbit boundary. Taking advantage of the left/right symmetry of the FWMAV to eliminate redundancy, we can omit half of the spherical gusts corresponding to the local FWMAV body coordinate shown in Figure 1 ($\boldsymbol{B_Y} < 0$).

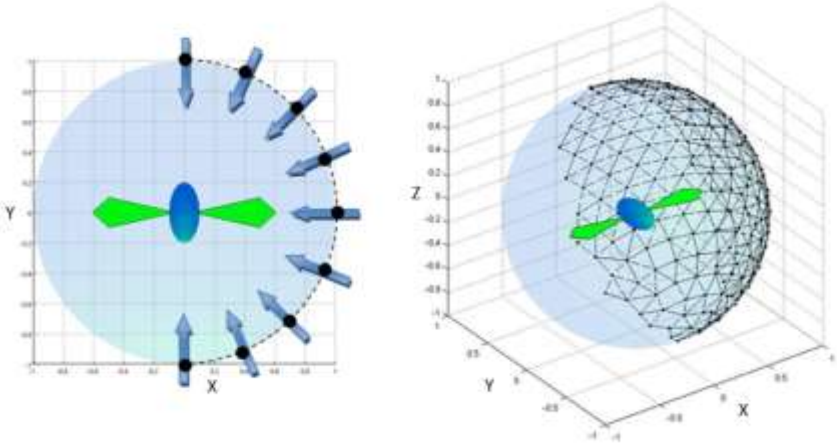


Figure 5. Lateral (left) and spherical (right) gust vector distributions

F. Optimization

For this research, the method of moving asymptotes (MMA) gradient based optimization originally presented by Svanberg [18], is coupled with the flapping wing and wind gust models. The optimization approach includes single gust and multiple gust profiles for mean efficiency optimization. In each optimization case, minimization of the cycle average power over the entire gust simulation is used as the objective function. The peak control power expended and the total flight orbit spherical displacement of the FWMAV are used as metrics for constraint behavior.

Given the design parameters $x=[x_1,\dots, xj_{=n}]^T$, implicit constraint behavior $f_i(x)$, objective function $f_0(x)$, and their gradients $\Delta f_i(x)$ and $\Delta f_0(x)$, the MMA obtains an optimal solution for the given design iteration from a convex, separable sub-function in which the implicit constraint behavior is replaced with an explicit approximation. The MMA is unique in that the explicit constraint behavior is obtained as a linearization of the implicit constraints based on upper (U_j) and lower (L_j) asymptotes for which $L_j<x_j<U_j$. The magnitudes of the asymptotes are modified according to the behavior of the problem gradients. As an example, if the optimization process is oscillatory, the asymptotes may be squeezed closer to design parameters; conversely the asymptotes may be relaxed if the optimization is too steady. A more detailed discussion related to the theory and applications related to the method of moving asymptotes can be found in reference [18].

As part of the philosophy for the optimization presented, emphasis was placed on obtaining a reduction in power, but not at the cost of “run-away” control authority. If one were to simply measure the cycle average forces obtained from the baseline kinematic perturbations during trim and linearization, the additional forces incurred from the perturbations caused by control gain would be discounted. Therefore the behavior constraint related to peak control cost was added in which the control cost is calculated as the product of the linear aerodynamic power and changes in kinematic perturbations resulting from the controller gain. The constraint behavior related to the peak control cost is then a function of the cycle average power and a constraint coefficient such that the peak control cost is less than kP, where the constraint coefficient k represents the ratio of total cycle average power dedicated to changes in control input. The peak control cost is then monitored and captured by a peak detector over the entire gust simulation. The motivation behind constraining the peak control power stems from the practical power requirements for FWMAVs. In real work applications, fuel cells have not only limited capacity, but also limited discharge rates. An optimal design solution which requires low peak power input inherits a wider design selection of available power sources.

The optimization workflow diagram is presented in Figure 6. The first step in the process is to define the initial conditions of the design parameters. The initial conditions are applied to the objective function, which includes the FWMAV trim and linearization, LQR design of the controller, and discrete linear gust simulation. From the objective function the objective value and constraint behaviors are determined; the constraint behavior may be defined from a single gust disturbance or an average of multiple gust disturbances. The gradients of the objective value and side constraints are then calculated these along with the design parameters, are normalized with respect the design parameter side constraints. The normalized information is provided to the MMA optimizer which determines the new optimized design parameters. The new design parameters are un-normalized and presented to the FWMAV objective function. The entire process is repeated until a convergence is declared by the MMA optimizer, a predefined maximum number of iterations have been met, or the user has decided the convergence is satisfactory and terminates the study.

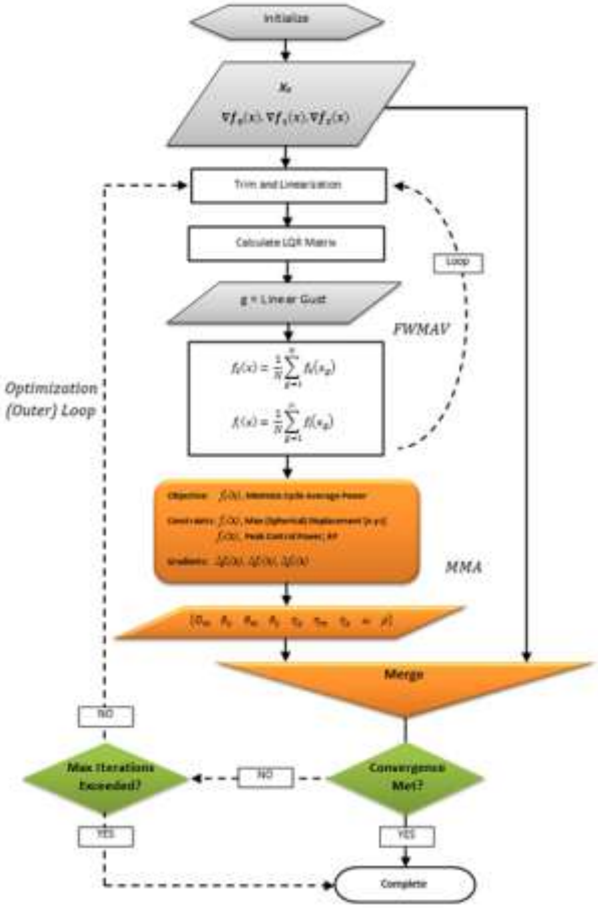

Initialize
x_0
$\nabla f_0(x), \nabla f_1(x), \nabla f_2(x)$
Trim and Linearization
Calculate LQR Matrix
g = Linear Gust
Loop
FWMAV
$f_0(x) = \frac{1}{N}\sum_{g=1}^{N} f_0(x_g)$
$f_1(x) = \frac{1}{N}\sum_{g=1}^{N} f_1(x_g)$
Optimization (Outer) Loop
Objective: $f_0(x)$, Minimize Cycle Average Power
Constraints: $f_1(x)$, Max (Spherical) Displacement (x, y, z)
$f_2(x)$, Peak Control Power; W
Gradients: $\nabla f_0(x), \nabla f_1(x), \nabla f_2(x)$
MMA
Merge
NO
Max Iterations Exceeded?
NO
Convergence Met?
YES
YES
Complete

Figure 6. MMA-FWMAV optimization flow chart for multiple gusts

The system presented is tightly coupled with each change in geometric and kinematic parameter affecting how the optimization will reform other parameters. With varying levels of influence, each of these changes affects all other aspect of the design objective and constraints behavior. The LQR coefficient ρ was introduced as the primary LQR parameter. Unlike other parameters which indirectly improve constraint behavior, ρ has the unique ability to balance the opposing constraints related to displacement and peak control power. The LQR coefficient fundamentally tailors the LQR solution to the given design optimization.

Kinematic parameters which are designated to the controller may be optimized with respect to their assigned baseline definition. The baseline defines how each particular kinematic parameter behaves in the absence of control authority and directly influences the formulation of the LQR gain matrix. For the purpose of this research, kinematic parameters assigned to the controller are excluded from the optimization. This was done to avoid the additional stochastic gradients which result from the direct changes to the LQR gain matrix.

For this study, the optimization can be applied to the FWMAV in the presence of a linear or nonlinear gust simulation. Both methods generate a closed-loop control based on LQR theory presented in the previous section. For the linearized method, cycle-average forces and power are calculated based on the state sensitivities calculated from the discrete time equation derived from the linearized periodic trim orbit. For the non-linear method cycle-average forces and power are derived from the perturbations calculated from applying the time marching method over the non-linear gust model.

G. Distributed Computing Utilizing SORCER Framework

This research was performed in a distributed computing environment using SORCER framework coupled with a “Multidisciplinary, Multifidelity, Model Based Computational Tool“(M^{3}CT). This approach was taken for two reasons: it mitigates the tedious process of managing multiple optimization studies and it accelerates the discovery of future solutions by encouraging the efficient reuse of existing models, applications, and configurations regardless of their native development environment.

SORCER provides an environment for which engineering models, applications, and data are made available across a potentially distributed, heterogeneous network of computing resources [4]. The SORCER environment is rooted in the Jini service oriented architecture technology developed by Sun [17]. SORCER inherits from Jini a federated service-to-service metacomputing environment that utilizes explicit leases, distributed events, transactions, and discovery/join protocols that enable SORCER to regard service hosts as network peers [12]. SORCER diverges from the Jini network service management by focusing on exertion-oriented programming and providing the execution environment for these exertions [12].

As a federated environment, SORCER permits a single service exertion (requestor) to organize a dynamic collection of collaborating services (providers) at runtime [13]. Each provider deploys a particular service (e.g. MMA or FWMAV model) by publishing its proxy object to the collection of SORCER registries as shown in Figure 7; this proxy object serves as a discovery mechanism between the provider and requestor. It’s through the registries that service requestors can dynamically explore and access each proxy object, given that the service availability is extended to that particular requestor by the provider; this exploration is performed without the requestor having any prior knowledge or dependency related to the provider’s platform, architecture, implementation, or network location. SORCER allocates the necessary computational resources for each exertion (or request) at runtime based on the requirements presented by the requesting service and the federation of providers. This metaprocessing allows the aggregation of the service requestor and providers, which can utilize the protocols defined in their respective proxy objects to dynamically collaborate with one another directly so as to appear as one program operating on a single machine [12].

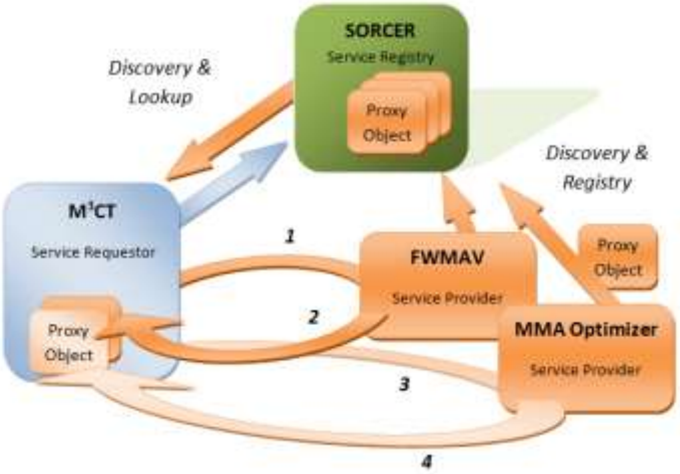
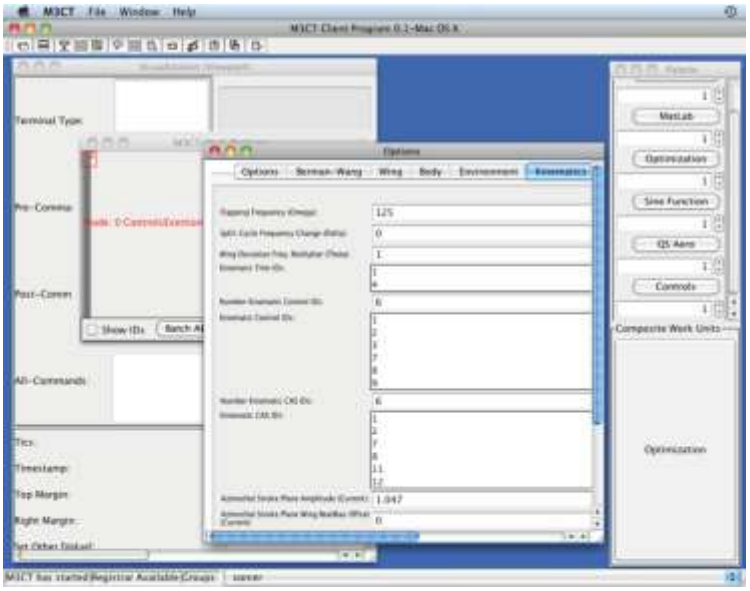


Figure 7. M^{3}CT and Provider Interaction with SORCER framework

In this research, the requesting service is the M^{3}CT graphical user interface, while the FWMAV models and MMA optimizer are implemented as service providers. The purpose of M^{3}CT is to improve access to the SORCER environment from an end-user's perspective. This is accomplished by providing a graphical tool suite, like the one shown in Figure 8, for which studies can be deployed using quick and intuitive initialization methods while providing autonomous monitoring for each case study. M^{3}CT accomplishes this within SORCER while circumventing the computer science background typically required for working in a distributed computing environment. Performing a study in M^{3}CT, using this research as an example, would require that the user simply select the FWMAV models and MMA optimizer which have already been identified in SORCER by M^{3}CT. Once the FWMAV and MMA modules are added to the workspace, the user simply defines the conditions for each model through the appropriate properties dialogue interfaces and then launches the study. M^{3}CT manages the context between the FWMAV models and the MMA optimizer while continuously monitoring the progress towards convergence. This process is depicted in Figure 7 above, in which M^{3}CT invokes the FWMAV model through the network to generate the baseline results (step 1). A completion is signaled to M^{3}CT from the FWMAV provider, which then processes the data (step 2) and invokes the MMA provider, providing the newly obtained data as the optimization conditions (step 3). The MMA provider then signals M^{3}CT that it has completed, and M^{3}CT retrieves the data from the MMA provider, which contains the next iteration optimization values to be passed to the FWMAV (step 1); this process is repeated until a predefined condition is met such as a convergence acknowledgment return from step 4.



125
0
1
1
4
6
1
2
3
7
8
9
6
1
2
7
8
11
12
1.047
0

M3CT has started Registrar Available Groups | sorcer

Figure 8. M^{3}CT service provider graphical user interface

The FWMAV and MMA applications were kept in their original form by compiling them as stand-alone executables using the Matlab Compiler™. The FWMAV models include the quasi-steady blade element aerodynamics model, the LQR based controls, and the gust models, all originally developed by Bhatia, et al. [3] and Stanford et al. [16]. These models were coupled under a single main routine which provides design parameter management and methods for requesting gradient calculations. The MMA optimizer was compiled as a separate stand-alone executable utilizing the core routines presented by Svanberg [18] with the addition of an integrated design parameter, objective, and constraint manager. These additions allowed the MMA provider to operate in the distributed framework in a more modular fashion.

The first step in integrating the existing FWMAV and MMA algorithms into SORCER required a SORCER service provider application wrapper to be developed for each application. The SORCER development environment provides a set of utilities, such as a SORCER service provider/requestor template generator, for integrating arbitrary applications into SORCER. The FWMAV related models and MMA optimizer were kept in their original Matlab® implementation and integrated into SORCER and M^{3}CT using the provided template generator. Data was exchanged between each provider object and its corresponding executable using an ad-hoc, file-based, context-management approach. In the file-based approach, a provider object first writes to the executable input file, then invokes the executable, and finally reads from the corresponding output file. The file-based approach represents a crude, but effective proof of concept for representing methods of integrating existing models developed in programs like Matlab with little development time required by the originator.

A comparison to determine the difference in computational cost associated with running the FWMAV model and MMA in their native Matlab® script and as stand-alone executables was performed. The results indicated negligible differences in wall-clock time between the two executions; in most cases it was less than one percent in computation time, favoring back and forth. Additionally, the computational overhead associated with SORCER was also deemed negligible with 99% of the computational time for the study being consumed by the execution of the providers themselves, including the high overhead of the file-based read-write methods embedded in the provider executables. Future versions of the service-provider implementation demonstrated here will utilize distributed shared memories, such as Sun JavaSpaces **Error! Reference source not found.**, to improve efficiency while leveraging on the existing engineering methodology. The primary efficiency in utilizing the shared memory stems from the basic access speed associated with system memory rather than disc memory, which is inherently more costly. The shared memory approach also improves efficiency by accessing only changes in relevant parameters, rather than passing (through the network) large files which rapidly accumulate data throughout the iterations. Additional methods of improvement may be made such as facilitating the computation of finite-difference based sensitivities with parallel computing to obtain the gradients or utilizing the adjoint method used by Beran et al. [1] rather than the direct analytical method used in this research. It is the opinion of the authors that the overall benefit of collaborating in a the distributed computing environment and the wide range of problems that may be studied by leveraging the development of others far outweighs the existing cost associated with its use.

III. Results

A. Parameter Initialization

For this research, the FWMAV optimization was considered only for maintaining hovering flight with prescribed spherical constraints proportional to the gust frequencies and magnitudes. The FWMAV orbital trim calculation for closed-loop control of the FWMAV includes all physical aspects including gravity. As a result, the force inputs associated with maintaining orbital position (including hovering) are inherited in the control design. This eliminates the need to prescribe the additional design constraints related to lift that were required in similar optimization studies performed by Bryson et. al [4] .

For each optimization case, the constant physical attributes such as the body mass, wing thickness, and other terms defined in Table 1 were chosen to be consistent with the morphological parameters of the common hawkmoth (Manduca sexta) model presented by Hedrick and Daniel [7]. The prescribed wing thickness was distributed equally along each chord section and assumed to remain constant throughout the optimization. Subsequently, the wing density was derived from the hawkmoth wing planform area of 989.4 mm^2, wing mass of 0.046 g, and the aforementioned thickness. Unlike the hawkmoth however, the wing chord distribution was reflected symmetrically about the un-swept mid chord line due to wing geometric limitations of the FWMAV model.

Table 1. Prescribed dimensions based on the morphological parameters of the Hawkmoth (Manduca sexta)

Model Parameter	Value
Body	
Mass (kg)	3.75×10^{-3}
Length (m)	4.66×10^{-2}
Radius (m)	6.00×10^{-3}
Wing	
Radius (m)	5.30×10^{-2}
Mean thickness (m)	3.00×10^{-4}
Density (kg/m^3)	1.56×10^{2}
Flapping frequency (rad/s)	8.50×10^{2}

Two sets of optimizations are presented in this paper; the first study focuses on evaluating optimization methods associated with single, laterally, and spherically distributed gust disturbances under prescribed kinematic parameters. The second optimization study considers the addition of kinematics as either design parameters or control parameters. For all cases, the gust disturbances occur at a constant frequency of 0.25 Hz, which occurs over the first 340 of the total 680 flapping cycles. Both data sets utilize the geometric design parameters related to the wing along with the linear quadratic controller cost function coefficient ρ in their optimization. In the first optimization study, single gust optimizations were performed to provide a basis for comparison against utilizing the calculated mean average objectives and constraints from multiple gusts scenarios. The three gust distribution solutions are then compared against each other. In the second optimization study, we consider optimization under spherically distributed gust vectors with the addition of the kinematic design parameters θ_s and η_s. We also evaluate the effects of assigning θ_m, θ_o, η_m, η_o as either design parameters or control parameters. In both optimization studies ϕ_m and ϕ_o are assigned as control parameters, while K_ϕ, K_η, and ω are held constant at 0.010, 0.010, and 534 rad/s (85 Hz) respectively. The following cases and their results (later summarized in Table 3 and Table 4) are presented as examples of performing MDO using the distributed framework SORCER. Initial values and side constraints related to each case are presented in Table 2:

1A. Optimization of the wing geometry and the LQR ρ coefficient under a single lateral gust disturbance.
1B. Optimization of the wing geometry and the LQR ρ coefficient under multiple lateral gust disturbances.
1C. Optimization of the wing geometry and the LQR ρ coefficient under multiple spherically distributed gust disturbances.
2A. Optimization of the wing geometry, LQR ρ coefficient, and the magnitudes, offsets, and phase shifts of the wing pitch and sweep under spherically distributed gust disturbances.
2B. Optimization of the wing geometry, LQR ρ coefficient, and the phase shift of the wing pitch and sweep under spherically distributed gust disturbances. Here, the magnitudes and offsets for both the wing pitch and sweep are not optimized, but rather assigned to the controller.

Table 2. Design parameter initialization and side constraints

Parameter	Description	Initial Value Study 1	Initial Value Study 2	Minimum Value	Maximum Value
Kinematics					
θ_m	Wing sweep amplitude (rad)	0	0	-π/4	π/4
θ_o	Wing sweep offset (rad)	0	0	-π/4	π/4
θ_s	Wing sweep phase shift (rad)	0	0	-π/2	π/2
η_m	Wing pitch amplitude (rad)	π/4	π/4	-π/2	π/2
η_o	Wing pitch offset (rad)	π/2	π/2	-π/2	π/2
η_s	Wing pitch phase shift (rad)	0	0	-π/2	π/2
Controller					
ρ	LQR coefficient (-)	0.001	0.001	$1\text{x}10^{-10}$	10.00
Geometry					
C_0	Root chord (m)	0.047	0.064	0.008	0.100
C_S	Span break chord (m)	0.047	0.026	0.005	0.100
C_N	Tip chord (m)	0.047	0.008	0.005	0.100
SBR	Span break ratio (%)	0.581	0.581	0.125	0.875

B. Convergence

For identifying the completion of each optimization, a simple design convergence criterion was calculated based on the natural log of the absolute change between two consecutive design iterations. The design is considered converged when the solution is less than the prescribed tolerance; otherwise a visual validation may be used to identify a satisfactory solution. In many cases the visual validation suffices when slightly oscillatory convergence is present. This minor oscillation typically results from the sensitivities related to the dynamic peak control power constraint, which has a highly sensitive correlation with the spherical displacement and objective power.

In the event a convergence is not met due to excessively oscillatory solutions, the optimization can be terminated automatically by exceeding a predefined maximum number of iterations set by the user. Most instances of oscillatory solutions (or convergence failure) can be attributed to the opposing constraints related to peak control power and spherical displacement. These constraints may be difficult to balance and the failure to do so may prevent any feasible solution from being met. A proposed method to address this issue is to implement a design constraint relaxation factor in the event an oscillatory solution is detected. This may be done by treating the convergence history as a signal waveform and analyzing the signal strength for a prescribed number of iterations. The side constraints may be relaxed when the signal strength exceeds some predetermined threshold.

A second method for addressing design convergence requires determining an appropriate optimization step size. The MMA step size is a division of the normalized design parameters and may be adjusted to improve convergence. A correlation between the number of optimization design parameters and the ideal step size was prevalent in this study; however a specific quantification was never determined. It was found that for most cases, a step size of 1% was ideal; this typically resulted in a slow, but steady convergence. In optimization cases defining less than five design parameters, it was found that the step size could be increased to values greater than 10%. In some instances, this increase in step size resulted in faulty floquet solutions in the trim and linearization portion of the FWMAV model. These instances usually occurred at the beginning of a study when arbitrary initial conditions were prescribed to tightly coupled design parameters. It was determined that beginning a study with a small step size for given set of arbitrary initial conditions and then increasing the step size as the solutions began to show signs of convergence helped speed up the optimization while maintaining a steady convergence trend. Subsequently, modifications to the floquet solver were made so that the initial conditions for the floquet solver in the trim and linearization were initialized based on converged floquet solutions from the previous iteration, given that only very small changes in parameters are applied between iterations. In most cases this drastically reduced the total number of iterations required to converge on a trim solution.

C. Results from Optimization Study 1: Analysis of Gust Vector Distributions

The purpose of the first study is to provide insight related to the influence various gust profiles acting on the FWMAV and how they will affect the optimization performance. It is expected that a single gust vector will result in an optimal design for that particular scenario and decay in performance as the gust direction vector deviates further away from the optimizer direction. Additionally, optimizations resulting from the statistical average of multiple gust vectors will generate more robust design solution across the gust spectrum; however they are not expected to outperform the cases which involve optimization for a particular gust direction. Results from the three cases evaluated in this study are presented in Table 3.

Table 3. Summary of results from optimization study 1

Case	1A	1B	1C
Cycle Avg. Power (W/kg)			
Initial Value	4702.40	4702.40	4728.60
Final Value	76.9600	76.8829	970.48
Improvement	4628.90	4625.50	3758.10
Max Displacement (m)			
Initial Value	0.6495	0.6495	1.3420
Final Value	0.6005	1.0582	2.00
Improvement	-0.0490	-0.4087	-0.66
Peak Control Cost (W/kg)			
Initial Value	20.4682	36.6378	70.35
Final Value	0.9689	5.9838	51.27
Improvement	21.4371	32.4046	19.07
Controller (final value)			
ρ	9.9817	9.9783	10.00
Geometry (final value)			
C_0 (m)	0.0075	0.0075	0.0155
C_S (m)	0.0122	0.0115	0.0235
C_N (m)	0.0117	0.0111	0.0365

SBR (%)	0.7842	0.7355	0.4719

In the first two cases of this study, lateral response lacks the direct counteraction against lift which is present in the longitudinal gust vectors. The negated lift and power required to compensate for such are in part opposing weights in the average calculation and therefore the negated longitudinal effects are, to some degree, minimized. It may be advantageous to perform optimizations considering only the quarter sphere to minimize the positive lift, which inherently tends to contribute the lowest additional control cost. The third case (1C) final solution resulted in a much larger max displacement and cycle average power than the other two cases. This can be attributed to the combination of the additional longitudinal component of the gust distribution and possibly to some degree the dynamic nature of the peak power constraint. The failure to reduce the total objective power in this case resulted in a higher peak power constraint, which may have failed to stimulate the optimization. The convergence history data from the study related to the minimization of power design objective and the constraint behaviors for peak control power and spherical displacement are shown in Figure 9

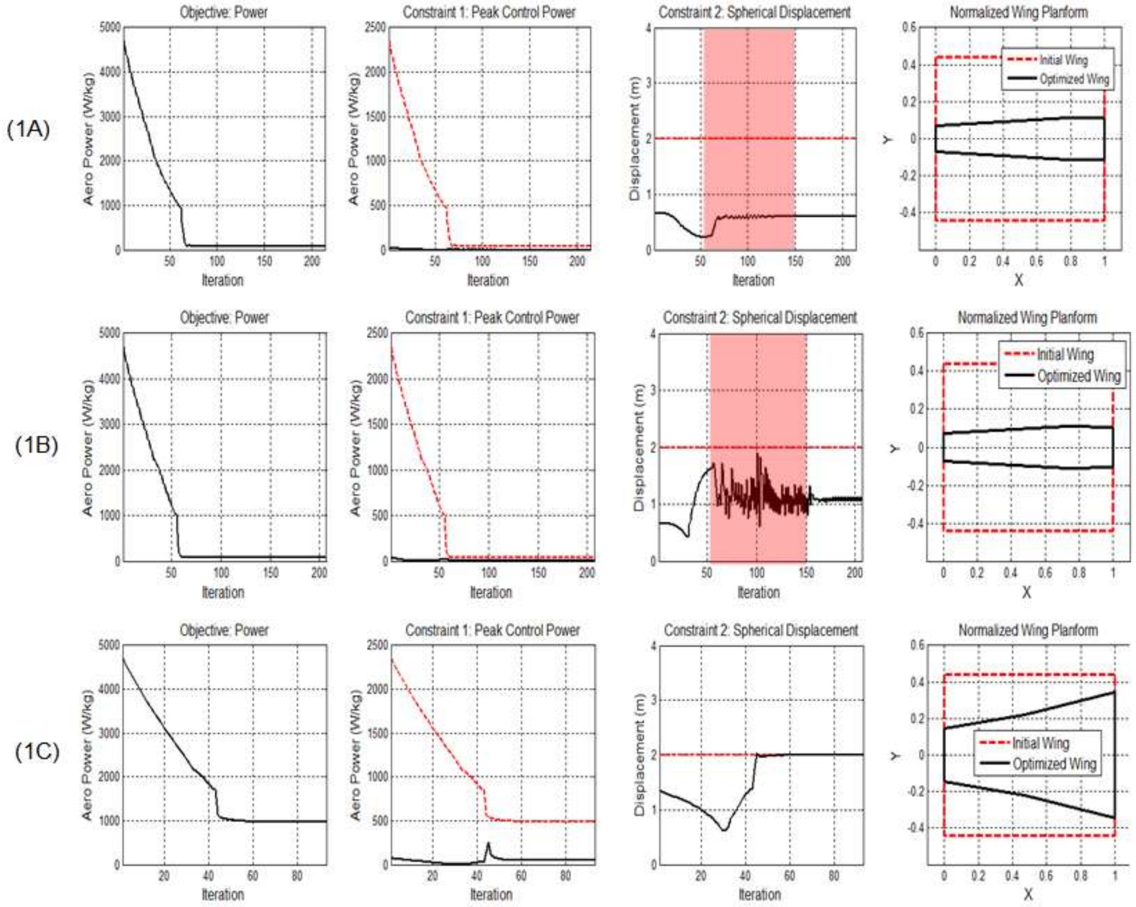


Figure 9. Convergence history of the design objective (left), constraint behaviors, and resulting wing planform (right) for (from top) cases 1A, 1B, and 1C.

For all three cases in the first study, an initial optimization step size of 1% was assigned along with a limit of 45 maximum iterations. The steps sizes were increased to 10% after all three cases failed to converge after the first 45 iterations; the cases were then allowed to continue for an additional 100 cycles. The third case (1C) was able to converge after 25 additional iterations, while the 10% increase for the first and second cases destabilized the convergence. The oscillatory convergence continued for the remainder of added iterations. For the third attempt, an additional 100 iterations were added and the optimization step sizes were reduced to 5%. The decrease in step size enabled both of the optimizations to converge on a steady solution after an additional 25 iterations (175 total iterations). The oscillations related to the increase in step sizes are highlighted in red for the spherical displacement in Figure 9 and for the design parameters in Figure 10.

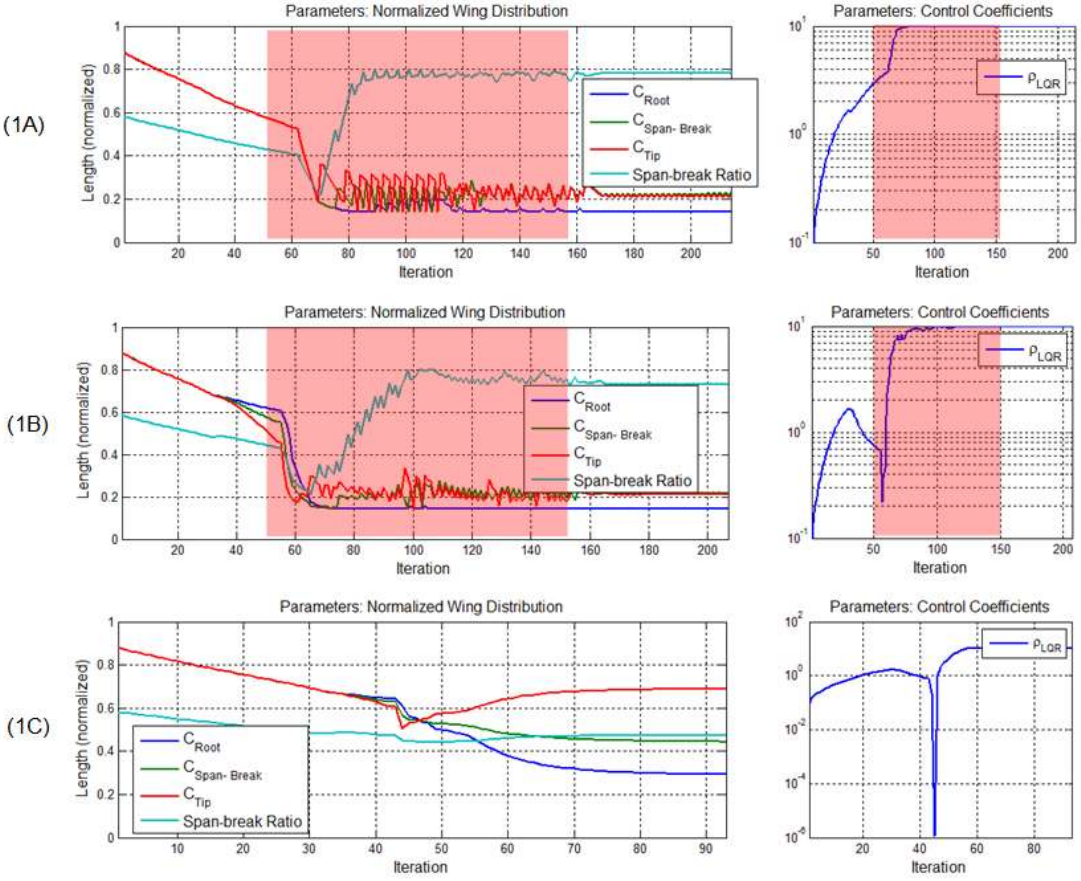
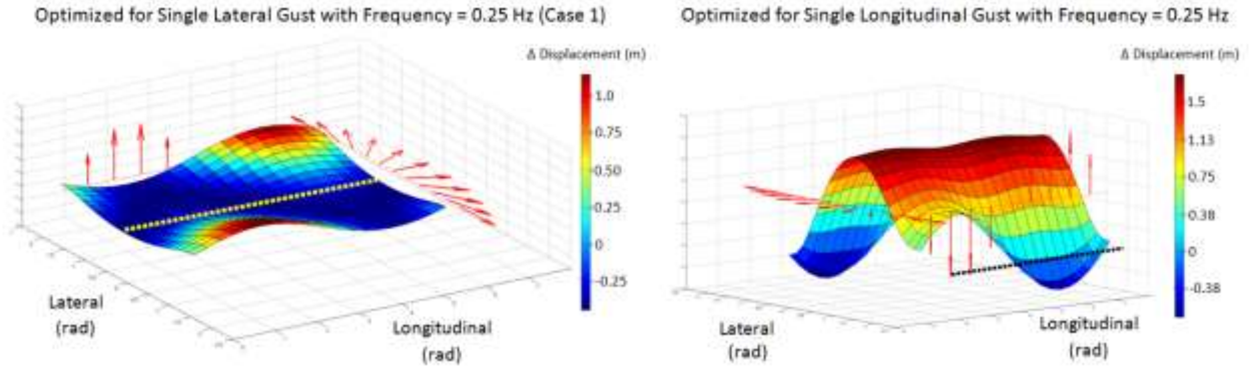


Figure 10. Comparison of the design parameters: chord distribution (left) and the LQR coefficient (right) for (from top) cases 1A, 1B, and 1C.

The maximum absolute displacement of the single gust optimization (case 1A) relative to the spherical gust optimization (case 1C), defined over an entire spherical gust distribution, are presented in Figure 11. While not presented as a case in this study, a single negative longitudinal gust optimization is shown (right) along with the single lateral gust optimization used in case 1A (left). The data is presented in the form of a three dimensional spectrum of the spherical displacement performance where the gusts are described using polar coordinates. As a guide, the red arrows indicate the lateral and longitudinal gust directions along their respective axis.


Optimized for Single Lateral Gust with Frequency = 0.25 Hz (Case 1)

Optimized for Single Longitudinal Gust with Frequency = 0.25 Hz

Figure 11. Gust spectrum change in displacement performance for single gust vector optimization and spherical gust optimization

The results from the analysis were consistent with our previous expectations for performing optimization over multiple gust profiles. In each instance, the single gust optimizations exceeded the third case in performance for the single lateral or longitudinal gust in which it was optimized for as indicated by the dark blue regions in Figure 11; a yellow line is shown along the strongest lateral gust component on the left. Conversely, the performance of both single gust optimizations decreased relative to the third case as the gust directions diverged from the respective lateral or longitudinal direction (as indicated by the dark red regions). As indicated previously, we see that the displacement reaches a maximum when the longitudinal gust force is directed downwards in the left of Figure 11. In the right we see that the lateral gust has a greater negative effect on the longitudinally optimized gust than the vertical gust has. Similar results can be found with regard to the control peak power and cycle average power, but are not presented here.

D. Results from Optimization Study 2: Analysis of Kinematics for Design and Control Parameters

The second study augments the previously developed FWMAV closed loop controller presented by Bhatia et. al [3]. Here we experiment to provide additional insight into the kinematic optimization as it relates to control by comparing cases in which a single degree of freedom is allocated to the flapping wing controller (case 2A) or all three degrees of freedom are allocated to the controller (case 2B). In each instance, kinematics corresponding to the degrees of freedom are either allocated to the controller or identified as design parameters for optimization. The cases are representative of the control input sets 1 and 2 used by Bhatia et. al [3] in their lateral and longitudinal gust disturbance analysis.

A summary containing the optimization results from the presented cases are shown in Table 4; the flapping stroke plane magnitude (ϕ_m) and offset (ϕ_o) are designated control kinematics and are never selected for optimization. For each of the presented cases, a spherical gust distribution containing 200 gust vectors was used. The cycle average power and maximum displacements were calculated as the mean average values over the entire 200 gusts. The peak control power was defined as the maximum of the peak powers obtained from each gust scenario. For each case an MMA step size of 1% was prescribed for each normalized parameter.

Table 4. Summary of results from optimization study 2

Case	2A	2B
Cycle Avg. Power (W/kg)		
Initial Value	3638.30	3670.40
Final Value	55.27	30.75
Improvement	3583.00	3639.70
Max Displacement (m)		
Initial Value	1.0869	0.0294
Final Value	1.9094	0.5445
Improvement	-0.8225	-0.5151
Peak Control Cost (W/kg)		
Initial Value	34.69	5329.20

Final Value	25.15	8.41
Improvement	9.54	5320.80
Kinematics (final value)		
θ_m (rad)	$9.31\text{x}10^{-4}$	Controller
θ_o (rad)	0.2584	Controller
θ_s (rad)	0.5868	0.0787
η_m (rad)	0.1256	Controller
η_o (rad)	1.3707	Controller
η_s (rad)	0.3995	-0.9115
Controller (final value)		
ρ	6.3283	8.1929
Geometry (final value)		
C_0 (m)	0.0206	0.0080
C_S (m)	0.0352	0.0117
C_N (m)	0.0371	0.0139
SBR (%)	0.6348	0.7457

From the data above, the second case in which all the kinematic parameter offsets and amplitudes assigned to the controller generated the most optimal design solution; surpassing the first case by nearly 180% in cycle average aero power, and reducing max displacement from case 1 by 350%. Equally as important is the reduction in peak control cost associated with the optimal solution in case 2 which is nearly one-third the control cost of case 1.

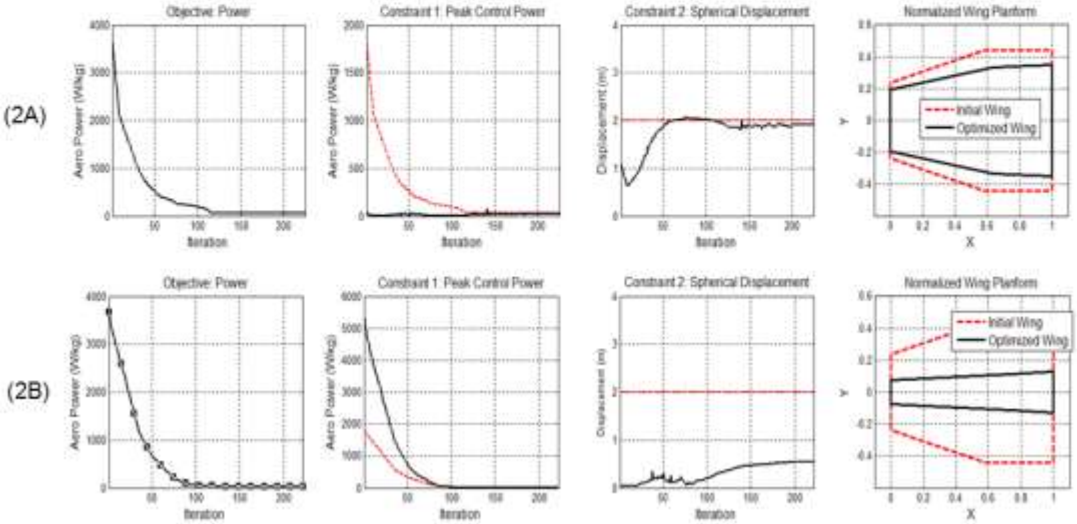


Figure 12. Convergence history of the design objective (left), constraint behaviors, and resulting wing planform (right) for (from top) cases 2A and 2B.

The peak power constraint in the second case benefited the greatest from this optimization as shown in Figure 12. The large reduction in wing planform area between cases 1 and 2 can be attributed to the need to reduce moments about the wing in order to assert higher control response. This geometry change likely contributed to the reduction in control power, but the linear quadratic cost coefficient, shown in the parameter convergence history in Figure 13, had the strongest influence on the control cost by sacrificing displacement as previously discussed in this paper.

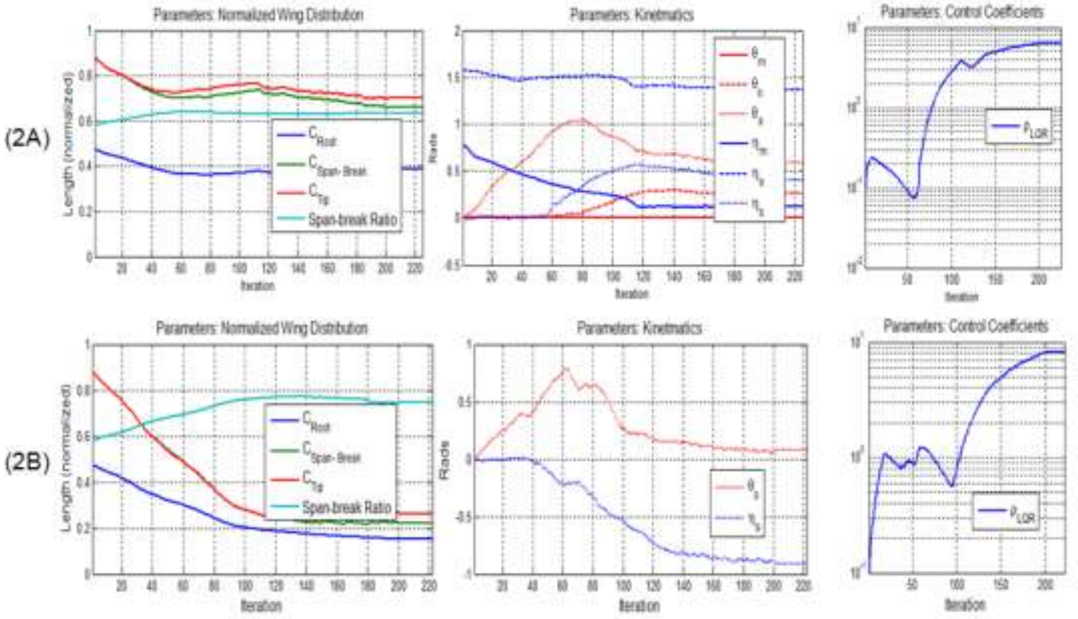


Figure 13. Comparison of the design parameters: (from left to right) chord distribution, design kinematics, and LQR coefficient for (from top) cases 2A and 2B.

To provide a basis for direct comparison, Figure 14 shows the orbital paths (left) and kinematic perturbations (right) when cases 1 and 2 are subjected to a single discrete lateral gust disturbance of 1 m/s over a 4 second time period followed by 2 second period with no gust. It can be seen from the orbital path that utilizing the state displacement as a design constraint provides only the magnitude at which the vehicle has been disturbed from its initial point; it does not require that the vehicle returns to that point. In case 1, we can see that the vehicle orbital path stochastically deviated further than that of case 2 and was unable to return to the point of origin in the allotted simulation time. The solution in case 2 generated a smooth orbital response path and was able to reasonably maintain its position at the point of origin in the allotted simulation time. The control authority distributions from both cases are shown on the right. The first case relied solely on the flapping stroke plane kinematic (ϕ) to maintain orbital stability requiring more mechanical work to return to return to its origin.

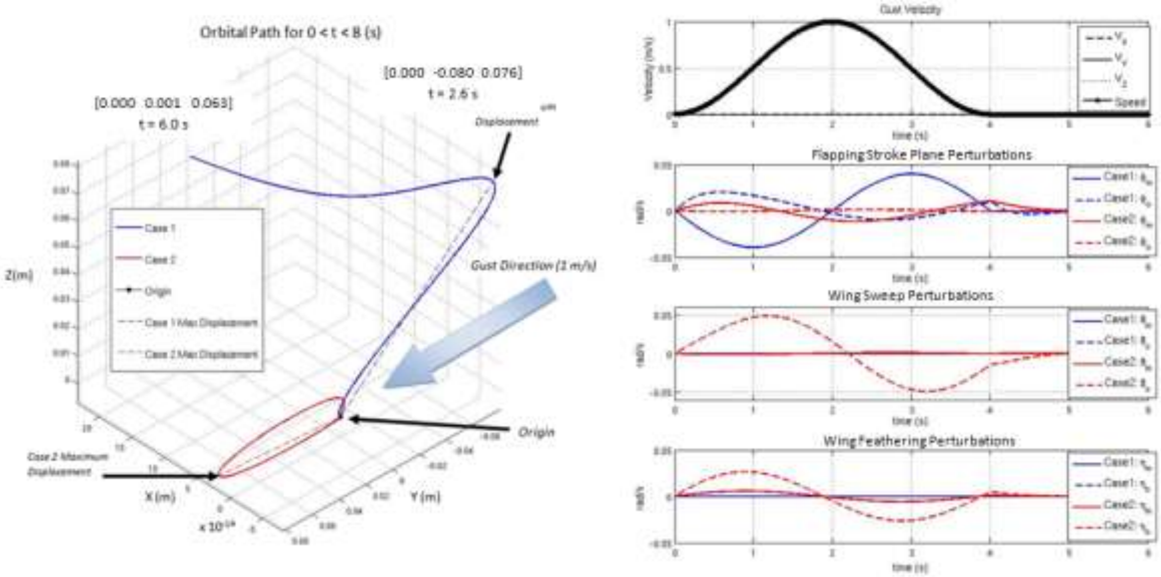


Figure 14. Orbital path (left) and kinematic perturbations (right) from a lateral gust disturbance imposed on the solutions to case 1 and case 2.

E. Multidisciplinary Optimization Cost in the SORCER environment

In comparing the computational overhead related to running the test cases in SORCER versus stand-alone, there was nearly no measureable performance cost associated with the addition of the distributive computing component. The typical run time for a optimization utilizing the linearized gust equations with 8 design parameters and 200 gust vectors was roughly four and half hours, while the optimizations involving nonlinear gust equations (not presented here) with the same parameters required up to 72 hours. Typical relative costs associated with running the linear versus the non-linear gust scenarios can be presented by the following equations which provide a rough time estimate in seconds, based on the number of gust vectors (N_G) presented, the number of design parameters (N_X), and optimization iterations (N_{IC}):

$$Linear\ Optimization\ Time = N_{IC}(40 + 0.001N_G)N_x \tag{13}$$

$$NonLinear\ Optimization\ Time = N_{IC}(40 + 2.5N_G)N_x \tag{14}$$

As can be seen in equation (13), little additional computational cost is associated with the number of gust vectors in the linearized gust model method; for example, to run 2500 linear gust vectors versus 250 vectors requires only 18 additional minutes. It was discovered however, that little optimization advantage was gained through additional gust vectors, so long as the vectors are equally distributed. Additionally, it has been verified, as shown in Figure 15, though example test cases that the linear and non-linear gust simulations provide consistent data with one-another in gust speeds up to 1.5 meters/second.

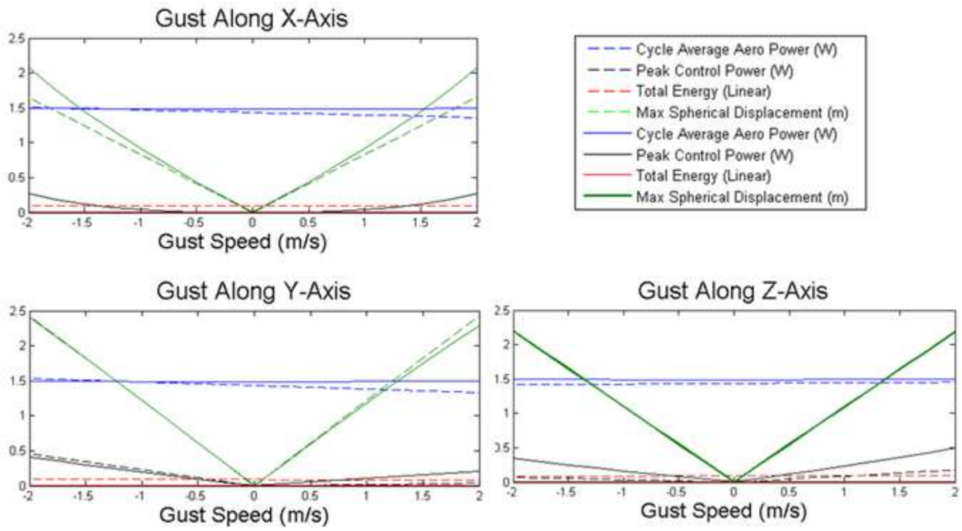


Figure 15. Linear gust versus non-linear gust for small disturbances

IV. Conclusion

A multi-disciplinary optimization study for a closed-loop flapping wing micro aerial vehicle was performed in a distributed computing environment utilizing the service-oriented framework SORCER. As part of the integration, a graphical user interface implementation of a service requestor (M^3CT) was coupled with SORCER. The M^3CT facilitates the research process by providing a tool for utilizing the distributed computing environment from a high level perspective while mitigating the tedious process of managing multiple optimization studies. This methodology promotes the use of the distributed framework by cloaking its implementation and inner workings. Utilizing this approach encourages the efficient reuse of existing models/applications regardless of their native development environment.

The required applications for this research were originally written in Matlab®. As part of the integration they were coupled in the SORCER environment by first, converting them to stand-alone executables and then deploying each as a separate service provider. The first service provider contains the qausi-steady blade element method, trim and linearization, LQR controls synthesis, and gust models along with an algorithm for providing gradients upon request. The second service provider incorporates the method of moving asymptotes for performing optimization. Each of service provider sub components may be modularized and re-introduced as separate service providers within SORCER to provider more versatility to the optimization design and provide encouragement to others who wish to expand upon the research. This modification would help exploit the most salient advantages of performing aerospace research and design in a distributive computing framework.

The test cases presented in study evaluated the kinematics, control, and wing shape optimization for the FWMAV with consideration for reduction in aerodynamic power. The optimization was performed under the constraints applied to both the orbital displacement caused by gust disturbances and the resultant peak control power. Using the MMA optimization method, multiple tightly coupled design parameters such as wing chord distribution, kinematic parameters, and control parameters were successfully demonstrated as viable design optimization considerations. It is evident throughout the test cases that the wing distribution optimization was complimented by optimal design changes in the kinematic parameters. Additionally, the linear quadratic cost function coefficient was successfully used to help balance the dynamic pair of opposing control authority and spherical displacement constraints.

The test cases analyzed thus far have provided encouragement to explore other design parameters for optimization; future cases will likely consider wing radius, baseline values related to controller assigned kinematics, and the split cycle parameter (δ). Additional modifications in the future will include developing autonomous methods for identifying appropriate optimization step-sizes which can balance computational overhead while avoiding oscillatory convergence. Future changes will also include exploring alternative statistical gust distributions for generating more representative performance values for a wide range of disturbances.

V. Acknowledgements

Thank you to Data Science Automation and Mr. Michael Robbeloth, funded by the AFRL Flapping Sciences Integration project under contract FA8650-04-D-B446, Dr. Michael Sobolewski of SORCERsoft, funded under contract FA8650-10-D-3037 0001, and Dr. Ray Kolonay of AFRL for their support in implementing the SORCER framework and M^{3}CT.

References

[1] Beran, P., Parker, G., Snyder, R., Blair, M., “Design Analysis Strategies for Flapping Wing Micro Air Vehicles,” *International Forum on Aeroelasticity and Structural Dynamics*, Stockholm, Sweden, June 18-20, 2007.

[2] Berman, G., Wang, Z., “Energy-Minimizing Kinematics in Hovering Insect Flight,” *Journal of Fluid Mechanics*, Vol. 582, pp. 153-168, 2007.

[3] Bhatia, M., Patil, M., Woolsey, C., Stanford, B., and Beran, P., “LQR Controller for Stabilization of Flapping Wing MAV in Gust Environments,” *AIAA: submitted for publication.”*

[4] Bryson, D.E., Stanford, B.K., McClung, A.M., Sims, T.W., Miranda, J.L., Beran, P.S., and Parker, G.H., “Multidisciplinary Optimization of a Hovering Wing with a Service-Oriented Framework and Experimental Validation,” *AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition Conference*, Orlando, FL, January 4-7, 2011.

[5] Doman, D.B., Oppenheimer, M.W., Sigthorsson, D.O., “Wingbeat Shape Modulation for Flapping-Wing Micro-Air-Vehicle Control During Hovering,” *Journal of Guidance, Control, and Dynamics*, Vol.33, No.4, pp. 724-739, 2010.

[6] Mamoud, Q.H., “Getting Started With JavaSpaces Technology; Beyond Conventional Distributed Programming Paradigms,” *Developer Technical Articles and Tips* [online database], URL: http://java.sun.com/developer/technicalArticles/tools/JavaSpaces/ [cited 01 June 2012].

[7] Hedrick, T.L., and Daniel, T.L., “Flight Control in the Hawkmoth Manduca Sexta: The Inverse Problem of Hovering,” *Journal of Experimental Biology*, Vol. 209, pp. 3114-3130, 2006.

[8] Oppenheimer, M.W., Doman, D.B., Sigthorsson, D.O., “Dynamics and Control of a Biomimetic Vehicle Using Biased Wingbeat Forcing Functions,” *Journal of Guidance, Control, and Dynamics*, Vol.34, No.1, pp. 204-217, 2011.

[9] Pennycuick, C.J., Klaassen, M., Kvist, A., Lindstrom, A., “Wingbeat Frequency and the Body Drag Anomaly: Wind-Tunnel Observations on a Thrush Nightingale (Luscinia Luscinia) and a Teal (Anas Crecca),” *The Journal of Experimental Biology*, Vol. 199, pp. 2757-2765, 1996.

[10] Pines, D.J., and Bohorquez, F., “Challenges Facing Future Micro-Air-Vehicle Development,” *Journal of Aircraft*, Vol. 43, No. 2, pp. 290-305, 2006.

[11] Rutkowski, A.J., Quinn, R.D., Willis, M.A., “Three-Dimensional Characterization of the Wind-Borne Phermone Tracking Behavior of the Male Hawkmoths, Manduca Sexta,” *Journal of Experimental Biology*, Vol. 210, pp. 37-45, 2007.

[12] Sobolewski, M., “Exerted Enterprise Computing: From Protocol-Oriented Networking to Exertion-Oriented Networking,”*On the Move to Meaningful Internet Systems: OTM 2010 Workshops, Lecture Notes in Computer Science*, Volume 6428/2010, 182-201, 2010.

[13] Sobolewski, M., "Metacomputing with Federated Method Invocation", Advances in Computer Science and IT, edited by M. Akbar Hussain, In-Tech, intechweb.org, ISBN 978-953-7619-51-0, s. 337-363, 2009.

[14] Sobolewski, M., “SORCER: Computing and Metacomputing Intergrid,” *International Conference of Enterprise Information Systems*, Barcelona, Spain, June 12-16, 2008.

[15] Stanford, B., Kurdi, M., Beran, P., McClung, A., “Shape, Structure, and Kinematic Parameterization of a Power-Optimal Hovering Wing,” *AIAA Structures, Structural Dynamics, and Materials Conference*, Orlando, FL, April 12-15, 2010.

[16] Stanford, B., Beran, P. S., Snyder, R., and Patil, M., “Stability and Power Optimality in Time-Periodic Flapping Wing Structures,” 53rd Structures, Structural Dynamics, and Materials Conference, 23 - 26 April 2012, Honolulu, Hawaii, Aug. 2011, pp. 1–13.

[17] Sun Microsystems, *Jini Architecture Specification*, v. 1.2, Sun Microsystems, Inc., Palo Alto, 2001.

[18] Svanberg, K., “The Method of Moving Asymptotes – A New Method for Structural Optimization,” *International Journal for Numerical Methods in Engineering*, Vol. 24, No. 2, pp. 359-373, 1987.