Rapid Development of Bespoke Sensorcraft: A Proposed Design Loop

Christopher Paulson∗ András Sóbester† Jim Scanlan‡

University of Southampton, Southampton, England SO17 1BJ, United Kingdom

The ability to quickly fabricate sensorcraft, or other small unmanned aircraft, via additive manufacturing techniques opens a range of new possibilities for the design and optimization of these vehicles. In this paper we propose a design loop that makes use of surrogate modeling and additive manufacturing to reduce the design and optimization time of scientific sensorcraft. Additive manufacturing reduces the time and effort required to fabricate a complete aircraft, allowing design iterations to be quickly manufactured and flight tested. Co-Kriging surrogate models allow data collected from test flights to correct Kriging models trained with numerically simulated data. The resulting model provides physically accurate and computationally cheap aircraft performance predictions. A global optimizer is used to search this model to find an optimal design for a bespoke aircraft. This paper presents the design loop and the progress made in implementing this design loop. Results are shown from Kriging models trained using numerically simulated data. Progress towards extracting aerodynamic data from flight testing small unmanned aircraft is also documented.

I. Introduction

Sensorcraft, or small unmanned aircraft with scientific sensor payloads, are required to carry a range of payloads and operate in environmental extremes. A majority of atmospheric sampling missions are currently completed using balloon launched, parachute recovered radiosondes. These systems are highly tested and reliable, but often drift far from launch sites making recovery impractical or impossible. Radiosondes have no method for active trajectory manipulation, making it difficult to collect data in a targeted region. Additionally, long distance data links limit the amount of data radiosondes can transmit to the ground.

Balloon released, autonomous glider systems improve the effectiveness of atmospheric sampling missions. A glider is capable of conducting targeted measurements and returning to the launch point for recovery. Since data can be stored on the aircraft and recovered at the end of the mission, much higher data acquisition rates are possible. They can carry a expanded array of payloads, making these platforms adaptable tools for atmospheric researchers. The aircraft can be reused many times and with quick turn around times between flights. Additionally, these sensorcraft can be operated from the ground if equipped with a small electric motor.

∗Graduate Research Student, Computational Engineering and Design Group, Faculty of Engineering and the Environment.
†Associate Professor, Computational Engineering and Design Group, Faculty of Engineering and the Environment, SMAIAA.
‡Professor, Computational Engineering and Design Group, Faculty of Engineering and the Environment, AIAA lifetime member.

Downloaded by Andras Sobester on January 16, 2015 | http://arc.aiaa.org | DOI: 10.2514/6.2015-0135
Our research focuses on the creation of a design loop to reduce the design and optimization time of a scientific sensorcraft. These sensorcraft often require a bespoke payload for a specific mission. Because payloads come in a variety of weights, sizes, shapes and power requirements, each payload would, ideally, require a bespoke aircraft. This work focuses on combing numerically simulated data and free-flight performance data into a single model that can be used to optimize designs for additively manufactured sensorcraft.

Three key areas are being explored during the development of this design loop:

1. Investigate how multi-fidelity data, from simulated and experimental sources, can be integrated into a single, persistent model capable of accurately predicting flight performance characteristics of small unmanned aerial vehicles.

2. Explore the coupling of parametric design, search and optimization techniques with digital additive manufacturing for the production of bespoke aircraft.

3. Test methods for extracting free-flight aerodynamics performance data from small unmanned aerial vehicles using low-cost flight management units.

II. The State of Additive Manufacturing of Aircraft

The University of Southampton’s SULSA (Southampton University Laser Sintered Aircraft), see Figure 1(a), was the first aircraft to be manufactured via Selective Laser Sintering, an additive manufacturing technology [Marks, 2011]. The aircraft was fabricated in five components and included kinematic joints and locking mechanisms, eliminating the need for fasteners. The SULSA aircraft was a technology demonstrator for additive manufacturing technologies in airframe fabrication. The lessons learned during the design and fabrication have since been integrated into later airframe designs. One of these designs is the 3.5 meter wingspan Spotter aircraft, Figure 1(b). Additive Manufacturing, specifically selective laser sintering, was extensively used in the fabrication of the fuselage, wings and empennage.

Since SULSA, a number of additional aircraft have been manufactured using Fused Deposition Modeling, or FDM, techniques. Aurora Flight Sciences demonstrated a 1.6 meter wing printed through an FDM process [Aurora Flight Sciences, 2012]. The MIT Lincoln Lab demonstrated a variable span unmanned aircraft wing fabricated using additive manufacturing [Stern and Cohen, 2013]. More recently, the University of Sheffield’s Advanced Manufacturing Research Centre demonstrated a blended wing body glider and powered blended wing body aircraft fabricated with FDM equipment [Nicholson, 2014].

Additive manufacturing provides a number of benefits over other manufacturing techniques for aircraft production. It is a digital manufacturing technology, meaning that each item can be unique at no added cost. Additionally, depending on the additive manufacturing technology, complexity is free. Additive manufacturing can produce components that are extremely difficult to produce with subtractive manufacturing processes.

Additive manufacturing is a relatively nascent manufacturing technology, which means materials research and software development are ongoing [Wohlers and Associates, 2013]. As advances are made in the field of additive manufacturing, there will be even more compelling reasons to consider the technology for sensorcraft fabrication.
III. The Design Loop

In this section, we propose a design loop for small sensorcraft. The goal of this loop is to predict the optimal airframe for a specific payload. This is the inverse of many unmanned aircraft system design processes today, which prefer integration of the payload into a fixed airframe. Our approach is made possible by the manufacturing flexibility offered by additive manufacturing and novel modeling and optimization techniques.

In engineering optimization, analysis functions are frequently computationally expensive, that is to say, these calculations take significant time and resources. This restricts the ability to use global stochastic optimizers to solve optimization problems, as many data points need to be evaluated.

In order to successfully use stochastic optimization methods, there must be a sufficient computational budget for the direct evaluation of each individual data point or the computational cost of an analysis function must be reduced. Surrogate modeling is a method to accomplish the latter. A trained surrogate model provides statistical estimates for the value of a function where no data is available from direct analysis. The estimate is generated using a carefully selected set of training data and mathematical models to understand a design domain [Krige, 1951, Matheron, 1963, Sacks et al., 1989, Jones, 2001, Forrester et al., 2008]. Additionally, data from computationally cheap and computationally expensive sources can be combined into a single model, a process known as co-Kriging [Kennedy and O’Hagan, 2000, Forrester et al., 2008]. The added flexibility of merging multi-fidelity data into a single model leads to a further reduction in the computational resources required to build a surrogate model.

The proposed design loop is based on co-Kriging models trained with numerically simulated data and data captured during free-flight testing. Methods for extracting data through flight testing are still being developed, as described in Section IV, so the results presented here rely solely on numerically simulated data.

Each aircraft design takes approximately 12 to 16 hours to numerically simulate, and between 24 to 72 hours to fabricate and fly an actual aircraft. Co-Kriging excels at producing predictive models with limited data, making it a useful technology for cases where data point evaluation is costly. Additionally, Co-Kriging models consolidate multi-fidelity data efficiently; the model’s accuracy significantly improves with the addition of limited high-fidelity data. The resulting model can provide a prediction at any point in the domain, usually in milliseconds, enabling global searches with stochastic optimizers. In this work, low-fidelity data refers to numerically simulated results, while high fidelity results are generated by free-flight testing the aircraft.
This section will start by describing the optimization problem to be solved. This will provide insight into the remainder of the section, which is dedicated to geometry generation, numerical simulation, Kriging model training and the results from the application of a multi-objective optimizer. A summary of the optimization process is shown in Figure 2.

Figure 2. A graphical flow chart of the design loop optimization process. Each optimizer (blue nodes), surrogate model (light yellow nodes), analysis or constraint function (red nodes) can use data available in the column and makes the result available along the row. Execution of the process begins at the optimizer and progresses down each row. The optimizer is responsible for generating a design vector, \( x \), which is then used to make all necessary evaluations. The results are made available to the optimizer once the evaluation is complete and a new design vector is generated. This figure follows the XDSM guidance as outlined by Lambe and Martins [2012].

### Optimization

The design loop returns a set of aircraft that best satisfy particular mission criteria. To identify this set of designs, a constrained multi-objective optimization problem is created and solved. Equation 1 summarizes the optimization formulation used in this design loop.

\[
\begin{align*}
\text{minimize} & \quad F_{\text{drag}}(x), \alpha(x) \text{ and } v_{\text{airspeed}}(x) \\
\text{such that} & \quad F_{\text{lift}}(x) - F_g(x) = 0, \\
& \quad b_{\text{max}} - b(x) > 0, \\
& \quad Re(x) - Re_{\text{min}} > 0, \\
& \quad \alpha_{\text{max}} - \alpha(x) > 0, \\
\text{subject to} & \quad x_i \in [x_i^L, x_i^U] \quad i = 1, 2, ..., n.
\end{align*}
\]

Three objectives are explored: minimizing the aircraft’s drag force at cruise, minimizing the aircraft’s angle of attack at cruise and minimizing the airspeed of the aircraft. An equality constraint on the lift force of the aircraft is used to ensure the aircraft is operating in steady level flight. Three inequality constraints are placed on this optimization: the span must be no greater than 1.5 meters\(^a\), the chord Reynolds number

\(^a\)A constraint imposed for logistical reasons, primarily aircraft transportation.
at cruise must be greater than 70,000\(^b\) and the angle of attack, \(\alpha\), must be small such that the optimizer does not design an aircraft that operates near the stall point of the wing. As the aircraft is operating in low Reynolds number conditions, the maximum allowed angle of attack is \(\alpha_{\text{max}} \approx 10^\circ\).

Before applying this optimization problem to the co-Kriging models, a number of steps must be taken. First, a robust geometry parametrization is required to create geometries for simulation and fabrication. Next, those geometries need to be numerically simulated. Finally, the data for the numerical simulations are used to train Kriging models. These topics will be discussed in the next three sections, with results of this optimizer action on the trained model presented in the fourth.

**Aircraft Geometries**

The method of defining an aircraft geometry is fundamental to this, or any, design and optimization process. A *geometry* is a parametric description of a physical shape. This description can take many forms, such as: a computer aided design file, a mathematical description of a shape, a configuration file that describes the dimensions and orientations of shape primitives, or a script that dictates how a geometry should be formed.

Sőbester [2014] recommends that the description of a geometry should be concise, robust and flexible. A parametrization scheme should specify a shape with as few variables as possible, create a geometry that responds sensibly to changes in the design description, and can be used to create complex shapes or combinations of shapes.

The geometric description used in this paper is based on the work of Sőbester [2014] and Sőbester and Forrester [2014], specifically the publicly available AIRCONICS code\(^c\). Rhinoceros 5 [McNeel], built on the OpenNurbs toolkit [OpenNurbs], is used as a CAD package for geometry creation. Rhinoceros 5 is a powerful tool for generating aerodynamic surfaces. It provides computer aided design tools that can be used to generate the required fabrication file, of the STL format, for use in additive manufacturing equipment.

The geometry code used to generate lifting surfaces implements three suggestions from Sőbester [2014]: it uses object-oriented shape primitives, separates the shape definition and scaling properties of geometries and the use of a leading edge geometry-attached curvilinear coordinate system to define the shape of the geometry.

Consider an axis that is anchored at the root chord and the tip chord of a wing and is coincident to the leading edge of the wing. This axis, known as the \(\epsilon\) axis, has a non-dimensional value of zero at the root chord and one at the tip.

A lifting surface geometry is generated by defining values of sweep, dihedral, twist and airfoil section as functions of \(\epsilon\). Sweep and dihedral define the shape of the leading edge of the wing, while twist and chord define the shape of the trailing edge. The airfoil cross section is also defined at each point along the \(\epsilon\) axis, defining the volume of the wing. The description of the wing can be summarized as:

\[
\text{Lifting Surface} = G(f_{\text{sweep}}(\epsilon), f_{\text{dihedral}}(\epsilon), f_{\text{twist}}(\epsilon), f_{\text{chord}}(\epsilon), f_{\text{section}}(\epsilon)).
\]

Each function \(f\) must be smooth and continuous for \(\epsilon \in [0, 1]\). The function can take any form, from a constant value to a piecewise function. This allows for a broad range of geometries, from rectangular wings to box wings.

Two additional parameters control the scaling of a non-dimensional wing fully dimensioned wing: a

---

\(^b\)A constraint based on empirical observations of a critical Reynolds number for airfoils by Carmichael [1981].

\(^c\)Please see [http://www.aircraftgeometry.codes](http://www.aircraftgeometry.codes) for more information. Last accessed 12/1/2014.
scaling factor applied to the entire geometry and a scaling factor applied to the chord length, called the chord factor. These scale the geometry whilst preserving the shape of the wing.

Wing properties, such as span, projected area, wetted area and aspect ratio, may be used as design variables in wing optimization. The relationship of the chord and scale factor to these metrics depends on the mathematical description of the wing shape (Equation 2). While an analytical approach may exist for some geometries, a generic approach of using a local search to identifying these parameters is used in this work.

(a) A tapered wing example.  
(b) A more complex boxing example.

Figure 3. Example geometries produced with Rhino and the parametric wing formulation of Equation 2.

The Python programming language is used internally in Rhinoceros 5 to generate geometries, allowing external libraries to be imported for use. The local search for the chord factor and scaling factor is conducted using the optimization toolkit of the SciPy package [Jones et al., 2001]. A Nelder-Mead simplex algorithm is used to modify the two scaling factors for the wing to obtain the target aspect ratio and wing area. The objective function for this optimization is an aggregate of two objective functions, one for area and one for aspect ratio. The optimal scaling factors produce a wing that is defined in both shape and scale. Two examples of parametrized wings are shown in Figure 3.

Rhinoceros 5 generates all necessary files for computer simulation and manufacturing. Additionally, the Rhinoceros 5 model allows for volume and mass calculations of the aircraft wing. These data, along with the files created, are stored in a database for later access.

For the example design loop results presented in this paper, the box wing architecture was explored. This aircraft is parametrized with three variables: aspect ratio, planform area and the sweep angle of the wing. Table 1 lists the ranges for each of these variables used for simulation.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Symbol</th>
<th>Units</th>
<th>Upper Limit</th>
<th>Lower Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspect Ratio</td>
<td>AR</td>
<td></td>
<td>4.5</td>
<td>2</td>
</tr>
<tr>
<td>Area</td>
<td>S</td>
<td>$m^2$</td>
<td>.65</td>
<td>.1</td>
</tr>
<tr>
<td>Sweep</td>
<td>$\Lambda_{s}$</td>
<td>°</td>
<td>45</td>
<td>15</td>
</tr>
</tbody>
</table>

Table 1. Box wing design parameters with upper and lower limits.

Numerical Simulation of an Aircraft

Once the digital geometry has been generated, the aerodynamic merit of the design is evaluated. This is done using two classes of tools: computational analysis and flight testing. The discussion of free-flight analysis methods is found in Section IV.

Computational fluid dynamics (CFD) provides the numerically simulated data for this co-Kriging model. Two CFD approaches for simulating small unmanned aircraft were considered: Navier-Stokes methods and Lattice Boltzmann Methods.
OpenFOAM\textsuperscript{d} was selected for the Navier-Stokes solver. A software package, droneCFD \textsuperscript{e}, was created to automate the simulation of parametric geometries and post-process the results. The only requirement to start a simulation is a digital file describing the geometry of the aircraft. droneCFD automatically configures the simulation directories, pre-processes the model, generates and meshes the simulation domain and runs the simulation. Command line arguments allow a user to specify many attributes of the simulation, including the angles of attack to evaluate and the airspeed of the domain fluid. droneCFD uses a steady state solver, so each simulation provides a numerical estimate for the lift and the drag of an aircraft.

This work also explores the use of Lattice Boltzmann methods (LBM) in place of Navier-Stokes methods. The main reason for looking at LBM methods is to reduce the complications of domain meshing. A commercial LBM code, Next Limit Technologies’ XFlow, has demonstrated the ability to accurately predict both lift and drag forces on an aircraft, results for a benchmark aircraft are shown in Figure 4.

![Figure 4. XFlow CFD and Wind Tunnel results for a reference aircraft. XFlow predicts both trends and forces acting on the aircraft.](image)

XFlow automatically refines the lattice during the simulation, automatically resolving wake regions in the simulation domain, demonstrated in Figure 5. Additionally, this refinement allows for the position and orientation of a geometry to be altered during the simulation, without user input. This simplifies the process of collecting drag polars from the simulation, reduces the number of simulations required and consumes less resources. Lattice Boltzmann methods are transient solver, resulting in a large amount of time dependent data. This complicates extracting a single value for use in a surrogate model, but provides a more realistic view of the fluid flow over the aircraft.

While both Navier-Stokes and LBM methods have demonstrated good performance numerically simulating sensorcraft, the results in this paper have been extracted from simulations completed with the OpenFOAM Navier-Stokes solver.

\textsuperscript{d}Please see http://www.openfoam.org/. Last accessed 12/1/2014.

Optimal Cruise Determination

The previous section discussed methods of numerically simulating fluid flow over an aircraft, but no mention was made of how those data would be used in the design loop. This section will present a simple aerodynamic model of the lift and drag of an aircraft. The choice of a simple aerodynamics model for the design loop is significant, as the data collected from numerical simulation must be directly comparable to the results collected from free-flight testing of the aircraft.

The model consists of two equations that describe the lift (Equation 3) and drag (Equation 4) of an aircraft as a function of angle of attack.

\[
L_f(\alpha, v) = q(m\alpha + C_{l_0}), \tag{3}
\]

\[
D_f(\alpha, v) = q(m_1\alpha^2 + m_2\alpha + C_{d_0}). \tag{4}
\]

The variable \(q\), or the dynamic pressure, is defined as:

\[
q = \frac{\rho v^2}{2}. \tag{5}
\]
This model depends on five variables, $m, C_{l0}, m_1, m_2$ and $C_{d0}$, which are unique to an aircraft geometry. The values for these equations can be obtained from numerical simulation of free flight testing. The results from these sources are directly comparable, and can be used in a co-Kriging model.

Once these model values have been found, the steady and level cruise configuration of the aircraft can be determined. In order for an aircraft to be at cruise, the lift force, $L_0$, generated by the aircraft must be equal and opposite to the weight of the aircraft. This condition is satisfied by a range of solutions. To find the most optimal solution, a second constraint is added, which is to minimize the drag forces acting on the aircraft.

Given that both $D_f(\alpha, v)$ and $L_f(\alpha, v)$ have smooth and continuous first partial derivatives, a Lagrange multiplier can be used to analytically find an operating point that minimize drag for a constant $L_0$.

$L_f$ will be used as the constraint function. Equation 3 can be expressed as an equality constraint by adding the $L_0$ term:

$$L_f(\alpha, v) = q(m\alpha + C_{l0}) - L_0 = 0$$

Given a function to minimize, $D_f(\alpha, v)$, and a constraint function $L_f(\alpha, v)$, a Lagrange multiplier is formed:

$$f(\alpha, v) = D_f(\alpha, v) + \lambda L_f(\alpha, v).$$

The minimum drag will occur at the critical points of $f(\alpha, v)$. Evaluating $\Delta \cdot f(\alpha, v)$ yields three equations:

$$\frac{\partial}{\partial \alpha} f(\alpha, v) = qu^2(2m_1\alpha + m_2) + \lambda mv^2 = 0,$$

$$\frac{\partial}{\partial v} f(\alpha, v) = 2qv(m_1\alpha^2 + m_2\alpha + C_{d0}) + 2\lambda qv(C_{l0} + \alpha m) = 0,$$

$$\frac{\partial}{\partial \lambda} f(\alpha, v) = 0.$$

Solving equations 6, 8, 9 and 10 with respect to $\alpha$, $\lambda$, and $v$ produces two real solutions for $\alpha$ and $v$:

$$\alpha = \left( \frac{A \rho (m_1 C_{l0}^2 - m_2 C_{l0} m + C_{d0} m^2)}{2 L_0 m_1} \right)^{\frac{1}{2}}$$

$$v = \left( \frac{A^2 C_{l0}^3 r^2 - m_2 A^2 C_{l0} m r^2 + C_{d0} A^2 m^2 r^2}{4 (4 L_0^2 m_1)} \right)^{\frac{1}{2}}$$

These analytical solutions can be verified and visualized by plotting Equations 3 and 4. Figure 6 shows two contour plots, one indicating the lift force and one showing the drag force. The lift force has a single black contour line, indicating a range of values that produce the correct lift for the aircraft (which is 12.25N in this example). The lift contour line is projected onto the drag plot. A second contour line shows a drag contour that is tangent to the lift contour, indicating a minimum value. The + indicates the analytical solution from Equations 11 and 12.

### Training Surrogate Models

The training process for a surrogate model begins with the design of a sampling plan. Each point in this sampling plan is described by a vector. The elements of these vectors represent the design variables of
the aircraft geometry. The ideal sampling plan is one that contains a number of sampling points that matches the available computation model, and that fills the design space uniformly. In this work, Morris-Mitchell optimized Latin hypercubes [Morris and Mitchell, 1995] are used. Each point in the sampling plan is evaluated using numerical simulation and some subset of the initial population is selected for fabrication and flight testing. The resulting data are used to train a co-Kriging surrogate model. The mathematical formulation and process of creating a co-Kriging model will not be discussed in this paper. The reader may wish to consult Jones [2001] and Forrester et al. [2008] for background and excellent discussions on Kriging models for engineering applications.

For the examples contained in this paper, an initial sampling plan was created with 60 points. The box wing aircraft is parametrized with three design variables: aspect ratio, planform area and wing sweep.

Once the geometries are created and evaluated, a number of surrogate models need to be trained. Five models are generated from the aerodynamics data obtained from post-processing CFD results, the coefficients from Equations 3 and 4. An additional three models are generated from data about the aircraft geometry: wingspan, mean chord length and airframe mass.

A Genetic Algorithm, specifically the NSGA-II [Deb et al., 2002], is used to identify the the correct hyper-parameters for each model. While the model could provide predictions at this point, additional steps can also be taken to improve the accuracy of the model. This is done by infilling additional points into the model.

The initial hypercube sampling plan was created to best fill the design domain, but had no information about the response of the model. Infilling a surrogate model with additional sampling points can be used to improve the accuracy of the model, specifically around features of interest. Kriging provides a toolset that allows for intelligent update strategies to improve the overall accuracy of the model. An accurate model is one that exhibits a low mean squared error over the entire domain. For the general improvement of a model, a search can be preformed that identifies regions of large error. When such a region is found, a sample point can be added and the model re-constructed. This process is repeated iteratively until the computational budget is exhausted or the model reaches the desired accuracy. For the models presented here, an additional 25 points were added to the simulation. Once the infill points are added, the model is retrained. The resulting eight models are shown in Figure 7.
Figure 7. Eight surrogate models trained with data from CFD and geometry generation. These surrogate models were used to produce the Pareto optimal set of designs shown in Figure 8.
Application of the Optimizer

Once the eight surrogate models are trained, the optimization problem introduced at the beginning of this section can be solved. The NSGA-II multi-objective optimizer from the PyOpt [Perez et al., 2012] Python package is used for this purpose. The optimizer is allowed to modify the three design variables, listed in Table 1, while respecting the allowed ranges for each variable. The design was constrained following the problem outlined in Equation 1.

The lifting force on the aircraft is determined by adding the mass of the payload to the predicted mass of the airframe. This required force is used to calculate the operating angle of attack and airspeed of the aircraft using Equations 11 and 12. The drag force acting on the aircraft, Equation 4, can then be solved. The span of the aircraft is directly predicted from a surrogate model, and the angle of attack is generated from Equation 11. The operating Reynolds number is calculated from values obtained from the root chord prediction surrogate model and from the velocity returned by Equation 12.

The Genetic Algorithm used a population of 40 individuals and ran for 500 generations. Figure 8 shows the Pareto Optimal, Rank 1, results for a number of different payload masses. Focusing on the 0.75kg payload condition, three designs were extracted. These aircraft are numerical results of these aircraft are listed in Table 2 and the aircraft geometries are shown in Figures 9 and 10.

Figure 8. A Pareto optimal set of designs for payloads of varying mass. The color of each data point corresponds to the mass listed in the legend.
Figure 9. A comparison between three Pareto optimal aircraft.

Figure 10. A top down view of the three Pareto optimal aircraft. Aircraft 1 is at the top of the image, Aircraft 2 is in the middle of the image and Aircraft 3 is at the bottom.
### Table 2. A summary table for three designs extracted from the 0.75 kg payload Pareto front. Figure 9 and Figure 10 show a comparison of these aircraft.

<table>
<thead>
<tr>
<th>Aircraft</th>
<th>Aircraft 1</th>
<th>Aircraft 2</th>
<th>Aircraft 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aspect Ratio</td>
<td>3.56</td>
<td>2.56</td>
<td>2.39</td>
</tr>
<tr>
<td>Planform Area</td>
<td>0.62 m²</td>
<td>0.21 m²</td>
<td>0.12 m²</td>
</tr>
<tr>
<td>Sweep Angle</td>
<td>17.77°</td>
<td>44.94°</td>
<td>40.49°</td>
</tr>
<tr>
<td>$F_{\text{drag}}$</td>
<td>1.99 N</td>
<td>1.48 N</td>
<td>1.42 N</td>
</tr>
<tr>
<td>Wingspan</td>
<td>1.48 m</td>
<td>0.73 m</td>
<td>0.53 m</td>
</tr>
<tr>
<td>$v_{\text{cruise}}$</td>
<td>10.76 ms⁻¹</td>
<td>15.73 ms⁻¹</td>
<td>19.69 ms⁻¹</td>
</tr>
</tbody>
</table>

### IV. Automated Flight Testing and Aerodynamic Performance Evaluation

This section discusses the techniques being developed to collect aerodynamics data from small unmanned aerial vehicles during flight testing. Because these data are directly compared to the aerodynamics models generated from simulated data, the type of the data collected must be compatible with the aerodynamics data gathered from CFD simulations. The complications of developing flight testing experiments can be reduced by creating a virtual environment where an aircraft can be directly controlled by a software implementation of the flight management unit, which is known as software-in-the-loop (SITL) testing.

### System-in-the-Loop Validation

The open source ArduPlane software [ard] is used for both SITL testing and free-flight testing. The SITL capability is built on the JSBSim JSB six degree of freedom flight dynamics model. The properties of an aircraft, including mass, geometry and aerodynamics properties, are defined in an XML document. These properties are loaded into the model and used to determine the physics of the aircraft in the simulated environment.

The virtual aircraft is controlled by a software implementation of the flight management unit. The Arduplane flight management software is written in C and is normally compiled to run on an ARM Cortex microprocessor. The code can be additionally compiled and executed on an X86-64 architecture. Instead of reading sensor data from hardware sensors, sensor values are read directly from the simulated aircraft. A connection is made to ground station software, allowing for monitoring and modification of virtual flights.

In addition to the ground station software, an additional software program is used to monitor the progress of the virtual aircraft and directly control it. This is called the auto-identification controller and is being developed as part of this thesis to conduct aerodynamic analysis and system identification of an aircraft. In the software-in-the-loop environment, this software communicates with the flight management unit over a UDP/IP connection. For free-flight testing, this code will run on an on-board single board computer \(^\text{f}\) that communicates with the flight management unit over a serial communication link.

### Glide Slope Testing

One measurement of lift and drag can be made by measuring the glide slope of an aircraft held at a constant pitch. This is the first test implemented in the auto-identification controller.

\(^\text{f}\)The specific single board computer being used is an ODROID-U2 single board computer with a 1.7 GHz quad-core ARM processor, 2 GB of LPDDR2 RAM and 32GB of eMMC flash memory.
An aircraft is given five waypoints: a takeoff location and four points that define a racetrack pattern. Figure 11 shows a representation of this flight path, along with the waypoint altitudes, specified as altitude above ground level (agl). During the descent leg, which is indicated by the blue segment, the auto-identification controller is triggered and records attitude and trajectory information. An example data set collected during this test is shown in Figure 12. Each color represents a different glide path with a linear regression fit to determine the mean glide slope. Based on the slope of the curve fit, or the mean glide slope, the variance of the glide slope is calculated. The mean pitch and pitch variance are calculated in the same way. This provides coordinates for the pitch angle and the mean glide slope, and a measure of uncertainty for each point.

Figure 12 shows the glide slope data plotted, where the data points and error bars are generated using the method described above. The ‘truth’ curve is shown in blue and is generated directly from the aerodynamic parameters used by the six-degree-of-freedom model. The green curve indicates the best fit curve through
The observed data. Initial guesses from the curve fit data come from the modeled aircraft parameters. Here, the initial conditions for the curve fit through the observed data were the parameters of the ‘truth’ curve. The final parameters of curve fit to data can be directly compared to the results of high and low fidelity numerical simulations.

![Graph](attachment:image.png)

**Figure 13.** System in the loop measured aerodynamic data

The discrepancy between the curves is under investigation. The likely cause of the difference is the flight management unit. In a pitch hold mode, the flight management unit may change pitch if the observed airspeed is too low or too high. This causes variance in pitch and changes the glide slope. At large pitch angles, an oscillating behavior emerges, where the flight management unit pitches down to build speed before pulling the aircraft back to maintain pitch. Several parameters in the flight management unit have been identified and will be tuned until the observed outcome of the SITL simulations matches the underlying aerodynamic model.

Because the autopilot may require tuning for each airframe, the scope of the auto-identification controller may be dramatically extended. This would include collecting a data set that allows for a complete digital model of an aircraft to be generated. This model would then be virtually flown in the SITL environment until the proper autopilot parameters are identified.

**Development Platform**

A development platform has been created to test the algorithms developed with the STIL testing on an actual aircraft. This platform is based on a commercial off the shelf RC aircraft. An autopilot, a single board computer and a number of sensors have been fitted to the airframe. A schematic layout of the aircraft can be found in Figure 15. The autopilot is a PX4 autopilot \(^6\) running the latest ArduPlane software\(^h\).

The autopilot has the ability to power on a small computer on the aircraft. This computer, when running, can take control of the aircraft and execute flight maneuvers to extract aerodynamic performance data. The

\(^6\)For information on the PX4 autopilot please refer to [https://pixhawk.org](https://pixhawk.org). Last accessed 12/1/2014.

\(^h\)Information on this software can found at [http://plane.ardupilot.com/](http://plane.ardupilot.com/). Last accessed 12/1/2014.
algorithms run on the single board computer are identical to the code run in the STIL environment. At any point, the pilot can regain control of the system using a radio controller on the ground. A significant amount of data is generated by the autopilot. The data from an example flight is shown in Figure 14. Real time data processing of these data will be sent to the ground in real time over a radio link. Once the hardware and algorithm are proven on this demonstration platform, it will be used in the aircraft fabricated for the design loop.

![Data Collection Graph]

Figure 14. A collection of data recorded during an autonomous flight of the test bed glider. These data show the fidelity of the information that can be used to extract the flight performance metrics from an aircraft.

V. Conclusion

A novel design loop that integrates numerically simulated data and free-flight data into a single model for optimization was introduced. Initial results from only numerically simulated data were presented. Progress on developing the algorithms and hardware for extracting aerodynamics information from aircraft flight testing was also introduced. Future work involves integrating flight data into a co-Kriging surrogate model and validating the design loop.
Figure 15. A schematic view of the components and sensors on the instrumented test aircraft.

Figure 16. The authors enjoying a flight test of the instrumented aircraft.
Acknowledgements

The authors acknowledge the use of the IRIDIS High Performance Computing Facility in the completion of this work. Additionally, the authors acknowledge Next Limit Technology for an extended evaluation license for the XFlow Lattice Boltzmann Solver.

References


