CAD-based shape optimisation of the NASA CRM wing-body intersection using differentiated CAD-kernel

Orest Mykhaskiv* and Pavanakumar Mohanamuraly † and Jens-Dominik Müller ‡
Queen Mary University of London, London E1 4NS, United Kingdom

Shenren Xu§ and Sebastian Timme¶
University of Liverpool, Liverpool L69 3GH, United Kingdom

In industrial design existence of a master CAD geometry of a product enables simultaneous multi-disciplinary collaboration. Adjoint CFD methods have become increasingly accepted for aerodynamic shape optimisations due to their low computational cost. However, use of CAD-based parametrisations for aerodynamic gradient-based shape optimisation is not widely used, one reason being that current CAD systems to do not compute derivatives. In this work, we present the automatically differentiated (AD) version of Open Cascade Technology (OCCT) CAD kernel which can provide derivatives with respect to CAD parameters. OCCT is differentiated in block-vector AD mode which significantly reduces the cost for computing the derivatives. This work contains further OCCT extension for NURBS-based optimisation with intersecting patches and a description of the surface mesh movement linked to the change of the intersection line. These techniques are applied to the drag reduction of the NASA Common Research Model via the modification of the intersection between the root fairing and the wing.

I. Introduction

The adjoint method allows rapid evaluation of the design sensitivity with respect to many design variables and thus is essential for efficient numerical aerodynamic shape optimisation with large design spaces. Traditional parametrisation of the design space has focused on producing bases with few variables that contain the most relevant modes. This restriction stems from a) the designer who prefers to work with parameters that map onto her/his intuition such as thickness or camber, and b) the limitation of non-deterministic optimisation algorithms to a very small a number of variables.

The adjoint approach offers a rethink: without a major penalty we can consider very rich design spaces that are automatically derived and contain all or nearly all possible modes. One option here is the node-based approach[1,2] which moves each mesh point and is guaranteed to be the richest design space the CFD mesh can represent. Since the CFD discretisation typically is transparent to highest-frequency oscillations, this space is actually too rich for the CFD and thus the gradients require regularisation.[3]

As an alternative, we consider here a CAD-based boundary parametrisation (BRep) which considers in the first instance all NURBS control points to be degrees of freedom. Typically BReps are fine enough to provide a rich set of controls over the surface,[4] but the highest frequency oscillatory modes are low enough to be adequately discretised by the CFD, hence precludes gradient regularisation. Automatic techniques for knot insertion or surface approximation with coarsened control nets can be used to adapt the design space.

The NSPCC technique[5] extends this approach from single to multiple patches with fixed edges and NURBS patch networks with arbitrary topology and order of geometric continuity between the patches. This

*PhD student, School of Engineering and Materials Science.
†PhD student, School of Engineering and Materials Science.
‡Senior Lecturer, School of Engineering and Materials Science.
§Research associate, School of Engineering.
¶Lecturer, School of Engineering.

is achieved by imposing continuity or other constraints at testpoints distributed along the patch intersections or on constrained features, computing the sensitivity of these constraint equations with respect to control point movement, and finally projecting the possible set of control point movements into the nullspace of the constraint matrix.

In this paper, we complement the NSPCC approach with another major component present in a typical CAD workflow: CAD models are usually created from several independent parts which are then combined with typical CAD Boolean Operations (Fuse, Common, Cut, etc.). In most cases, these operations perform surface-surface intersections to construct the final shape. As a result, the final CAD model includes several trimmed patches limiting design optimisation to the existing topology. To alleviate this, we include the recalculation of the patches intersection into our CAD-based design chain. This could be considered as a step towards greater integration of CAD into design optimisation loop, as we also devise complementary mesh movement technique that corresponds to the occurring changes in CAD topology.

An essential ingredient that enables CAD-based optimisation is the availability of parametrisation sensitivities (gradients), which quantitatively describe the influence of each design parameter on overall shape. These sensitivities are seldom provided by CAD packages and are mostly computed numerically. Robinson et. al[6] calculate CAD sensitivities of parametric CAD models from closed-source commercial CAD software using finite differences. The open-source CAD-kernel Open Cascade Technology (OCCT) was used as a geometric engine with sensitivities obtained either with finite differences or analytically[7] (for known simple shapes such as circles and cylinders defined by origins, radii and axes). For the NSPCC work[5] authors devise analytical derivatives and also use algorithmic differentiation in the compact in-house NURBS modeller. The recent incorporation of the surface-surface intersection sensitivities for the DLR F6 case[8] also relies on the finite difference method.

We alleviate the deficiencies of the finite differences using an algorithmic differentiated OCCT CAD-kernel.[9,10] The differentiated version of OCCT is equipped with necessary CAD derivatives which are exact. This software is suitable for a wide range of parametrisations and was applied for both parametric CAD model and NURBS-based optimisation[11] (based on an extension of the NSPCC approach). In the present work, we explore the intersection capabilities of OCCT augmented with the corresponding derivative information. When coupled with our in-house adjoint solver this presents the fully differentiated design chain, which we use to minimise the drag coefficient of the NASA CRM.

Maximisation of lift-to-drag ($\frac{L}{D}$) ratio is a major design goal for fuel efficient civil aircraft design. An increased $\frac{L}{D}$ ratio allows the aircraft to fly with higher payload for a given thrust, or fly with a smaller angle of attack with a given payload, for a longer time and with less fuel. The optimisation study for the NASA Common Research Model (CRM) wing configuration was performed for both cruise and off-design conditions,[12,13] as well as coupling of both the aerodynamic wing planform optimisation with the structural design using high fidelity methods. Aerodynamic shape optimisation of a wing-body-tail configuration subject to a trim constraint was investigated.[14] In all these cases the shape improvements were controlled by the free-form-deformation parametrisation both for the shape of the wing and the tail.

In this work, we investigate the NASA CRM wing-body configuration using the aforementioned CAD-based approach. We aim to redesign the wing root fairing and determine its optimal intersection with the fixed wing. The fairing movement in the contact zone with the fuselage is restricted. Both the fairing and the wing are modelled in CAD software as B-spline surfaces.[15] Similar parametrisation/optimisation techniques were used in author's DLR F6 work,[8] where the changes of the fuselage(in the fairing region) suppressed the wing-fuselage junction separation. Here we incorporate much-complicated CAD system and benefit from more general method valid in a case of a non-trivial intersection curves. The novelty of the proposed paper, therefore, lies in a) the propagation of the displacement of the intersection into the deformation of the surface mesh, and b) the inclusion of the derivatives due to the moving intersection in the overall design sensitivity. The efficient calculation of the derivative is achieved by using forward differentiated OCCT software in a forward block-vector mode.

This paper is organised as follows: Section II describes the coupling between the CFD solver and CAD kernel. It also provides the details on all necessary CAD ingredients: CAD differentiation, treatment of intersections, corresponding mesh movement and sensitivities followed by optimisation results in the Sec. III.

II. Optimisation Design chain

II.A. Discrete adjoint equation

For a typical CAD-based shape optimisation problem the following is usually considered:[16]

$$\min_{\alpha} \; J(U(X(\alpha)), X(\alpha), \alpha) \tag{1}$$

$$R(U(X(\alpha)), X(\alpha)) = 0\,. \tag{2}$$

The defined objective function J (usually the aerodynamic force like lift, drag, etc.) is minimised with respect to the CAD geometry with design parameters α. Here R defines the flow field (2) by system of Reynolds-Averaged Navier-Stokes equations, with the state variable U and a computational mesh coordinates X. The latter depend on design parameters α. For these problems the number of design parameters is large and one usually employs computationally efficient adjoint technique which could be derived by application of a chain rule to the system (1)-(2):

$$\frac{dJ}{d\alpha} = \Big[\frac{dJ}{dX} + \nu^T f\Big]\frac{\partial X}{\partial \alpha}\,, \tag{3}$$

where

$$f = -\frac{\partial R}{\partial X}\,. \tag{4}$$

Here ν represents the solution of adjoint equations:

$$\Big(\frac{\partial R}{\partial U}\Big)^T \nu = \frac{\partial J}{\partial U}\,. \tag{5}$$

After computing the solution of primal and adjoint equations (2),(5), one can rewrite cost function gradient in terms of surface grid points derivatives:

$$\frac{dJ}{d\alpha} = \frac{dJ}{dX}\frac{dX}{dX_S}\frac{dX_S}{d\alpha}\,. \tag{6}$$

The first term is the sensitivity of the cost function with respect to the internal volume mesh. The second term is the sensitivity of the surface to volume mesh deformation and the last term is the so-called CAD sensitivity.

II.B. Volume mesh deformation

The relation $X = X(X_S)$, describes the mesh morphing method employed to translate the surface mesh displacement into the volume. In the present work we used the inverse-distance weighted interpolation (IDW) to interpolate the surface displacements into the volume. We exactly differentiate the IDW mesh morpher to obtain the projected surface sensitivity from the volume sensitivity (also CFD sensitivity). The IDW is an explicit surface-to-volume interpolation described mathematically as,

$$\delta X_i = \frac{\sum\limits_{j\in\partial\Omega} \mathbf{W}(||X_i - X_S^j||)\delta X_S^j}{\sum\limits_{j\in\partial\Omega} \mathbf{W}(||X_i - X_S^j||)} = \sum_{j\in\partial\Omega} a_i^j\, \delta X_S^j = \mathbf{A}\delta X_S \quad i \in \Omega \tag{7}$$

Here $\mathbf{W}$ describes the weighting function based on inverse-distance ($||X_i - X_S^j||$), Ω and $\partial\Omega$ are the volumetric and surface domains. The above equation can be exactly differentiated (due to its linear form) to yield an explicit surface-to-volume sensitivity projection as shown in equation (8). Unlike other mesh deformation methods IDW can be implemented in a matrix-free form (both morpher and sensitivity projection) making the method quite efficient numerically. In addition, IDW is comparable in its final mesh quality to RBF and linear elasticity methods.

$$\left(\frac{\partial X}{\partial X_S}\right)^T = \Big[a_i^j\Big]^T_{n\times n_b} = \mathbf{A}^T \tag{8}$$

One must note that IDW is an explicit method, hence it is possible to obtain the surface sensitivity projections only for the desired surface nodes covering the CAD design surface. This brings down the computational time quite significantly.

II.C. CFD solver and sensitivity

The first term in (6), usually called *CFD sensitivity*, corresponds to the flow sensitivity w.r.t the volume grid points X. These derivatives can be calculated efficiently using CFD solvers that can obtain the adjoint solution of the primal flow problem. In this work we use our in-house discrete adjoint solver STAMPS (erstwhile *mgopt*).[17] STAMPS is based on vertex-centered finite volume scheme with MUSCL type reconstruction with approximate Riemann flux solver to achieve second-order spatial accuracy in the convective terms. The viscous source terms are obtained using a modified Greeen-Gauss formula. It uses a novel preconditioned Runge-Kutta multi-grid temporal solver called JT-KIRK.[18] The power of STAMPS lies in its pre-conditioning scheme, which itself is an ILU pre-conditioned inexact Newton-Krylov solver. JT-KIRK scheme has been shown to provide deep convergence in the primal solution, which is necessary to achieve a stable adjoint solution, even for mildly separated flows. The discrete-adjoint solver in STAMPS is a hand-assembled fixed-point loop using differentiated routines from the source transformation AD tool Tapenade.[19] The JT-KIRK scheme is re-used in the adjoint FPI by suitable transposition of the pre-conditioner and it guarantees that the iterative convergence rate of the adjoint code is identical to that of the linear code, and the asymptotic convergence rate of the original non-linear RANS solver.[20]

II.D. Differentiated Open Cascade Technology and CAD sensitivity

The third term (*CAD sensitivity*) represents the derivative of the surface grid points X_S with respect to the CAD model design parameters α. This term is calculated using the automatically differentiated version of OCCT.[9,10]

Open Cascade Technology is a powerful open-source CAD-kernel and highly amenable to automatic differentiation due to its source code availability. It provides almost all functionalities present in a modern CAD-system, including read/write support for popular CAD file formats, point inversion (point projections), NURBS modelling and surface-surface intersection algorithms which are extensively used in this work. The complete OCCT kernel was differentiated[9,10] using the AD tool ADOL-C[21](Automatic Differentiation by Overloading in C++). After injection of ADOL-C into native OCCT code (OCCT standard `double` types were redefined to use ADOL-C specific ones), the operators in the CAD software were overloaded to perform derivatives calculations. Due to the inherent complexity of OCCT, some additional code modifications were required for successful differentiation. These modifications were concerned with compilations problems and run-time issues arising during the testing phase. Still, this general type replacement allowed seamless integration of the AD tool into OCCT and as a result, the differentiated version of OCCT is capable of providing derivative information in almost any of the existing algorithms.

ADOL-C provides two options/headers for differentiation (traceless and trace-based), each determining a different implementation of the overloaded operators. The operators in the traceless version are designed to perform gradient computation simultaneously with the primal function evaluation, thus derivative information is propagated along the path of the program execution. Here two modes are supported: scalar (derivatives are calculated w.r.t. a single parameter) and vector (w.r.t. multiple parameters) both supporting only forward/tangent mode of AD. In the trace-based version, ADOL-C stores the internal structure of the algorithm to be differentiated (trace) and then computes derivatives based on the trace. As opposed to the traceless option, both forward and reverse/adjoint mode of AD are supported. The latter is computationally superior in the case of many program inputs (design variables). The differentiated OCCT can include both types of headers (supporting both forward and reverse AD), but in this work, we exploit only the traceless version.

OCCT modelling performance was affected by additional derivative computations present in the overloaded operators, but one could minimise this inefficiency by applying ADOL-C in the vector mode. Due to the optimisations in the AD software subroutines, the vector mode is faster than sequential scalar computations but requires more memory (scales linearly with the number of parameters). To ensure proper trade-off between the memory consumption and the computational efficiency, we subdivide our design parameters into several blocks and apply the vector mode AD to each of them. Alternatively, the reverse differentiation of OCCT could be also considered to minimise computational time. However, due to the complexity of parametrisations with intersections, the OCCT source code requires additional modification. Without these changes, effective trace generation for the intersection cases is not feasible. At this stage, we do not foresee the bottleneck in CAD sensitivities computations for gradient-based optimisation, where time spent on high-fidelity CFD calculations is dominant.

Therefore, any CAD-model parametrised in this version of OCCT, or alternatively BRep model read from CAD-vendor neutral file, is equipped with corresponding derivative data. The typical scenario for the CAD-based optimisation for a given CAD model starts with mesh point inversion (determining positions of the computational grid points in the parametric space of CAD surfaces), then derivatives are computed in these points. Calculation of CAD sensitivity w.r.t. some certain parameter requires its activation (seeding) by means of ADOL-C software, regeneration of the model or geometrical manipulation (e.g. computation of the intersection of two surfaces) with no difference to the non-differentiated original version of OCCT.

II.E. Gradient based optimisation

Finally, we assemble these consistent individual sensitivities from (3) using the chain rule to obtain the total sensitivity w.r.t the CAD design variables α. We then use this total sensitivity in the iterative gradient-based optimisation loop.

$$\alpha^{(n+1)} = A\big(\alpha^{(n)}, \frac{dJ}{d\alpha}(\alpha^{(n)})\big)\,, \tag{9}$$

In the above equation, A is a suitable gradient based optimisation algorithm. In the next sections, we describe the parametrisation and the influence of design parameters α on the intersection lines.

II.F. Parametrisation using NURBS

The geometry for the NASA CRM used in this work is provided at [URL: https://commonresearchmodel.larc.nasa.gov/]. We consider the NURBS-based redesign of the wing root fairing (the orange patch in Fig. 1). The other patches (in blue) i.e., the complete fuselage and the entire wing remain fixed and are not changed during the design process. Therefore the control points positions of the fairing surface compose our CAD design space.

Before starting the optimisation, we perform several pre-processing/re-parametrisation steps on the CRM geometry using standard tools available in OCCT. First, we remove the trim in the fairing patch located at the wing junction (Fig. 2). The resulting untrimmed surface is actually a re-approximation of the original one but does not contain a hole from the wing intersection. This step could be considered as a reverse-engineering problem since quite often to create valid and watertight CAD model one starts with several untrimmed surfaces and after application of CAD Boolean Operations they become trimmed (edges appear in the places of intersections). As shown in the Fig. 3, movement of control points of the resulting patch towards the wing will change the intersection line between the wing and the fairing. We allow the movement of the fairing inside the aircraft by extending the wing into the interior of the aircraft, therefore the valid intersection between these two parts always exists.

The original baseline design surfaces had a very dense distribution of control points (thousands) on the patches shown on the right of the Fig. 1. This type of surface is often a product of conversion from STL (surface tessellation) to the standard CAD file. From our previous experience, these parametrisations are not practical for NURBS-based optimisation as they could capture oscillatory modes introduced by generally non-smooth CFD gradient fields. This may result in the undesirable "wavy" surfaces, require high computational resources due to a large number of degrees of freedom and could stall the optimiser. On the contrary, the re-approximated fairing and the wing accommodate a much smaller amount of control points (only 22×22 for the fairing).

To maintain continuity between the design patch and the rest of the fuselage, the control points on the periphery of the fairing are not allowed to move, ensuring the fairing and the fuselage remain G0 continuous. Restriction of additional rows and columns of control points can also establish G1 or G2 continuity (based on the degree of NURBS). This manual step reduces the number of design parameters, and finally, we end up with (13×13) control points which are allowed to move in any direction, constructing 507 design parameters. Alternatively, one could employ the NSPCC approach, use all fairing control points but set the constraint on the fuselage-fairing boundary.

II.G. Dealing with intersection lines and corresponding surface mesh movement

Variation of the design parameters modifies the fairing surface, and hence the intersection with the fixed wing (Fig. 3). The technique to propagate and sync these geometrical changes with the computational mesh is therefore needed. Several works addressed this issue and were based on the derived relation between the initial mesh points that belong to the intersection line and the control points of the fairing.[8,22,23] In their

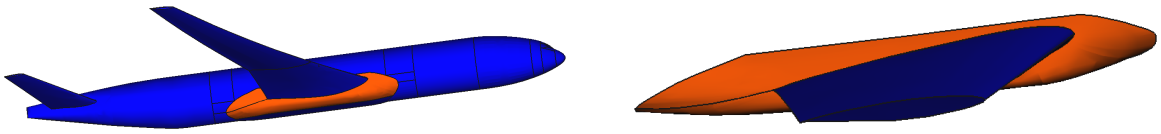

Figure 1. Left: B-spline surfaces forming the NASA CRM wing-body model; right: fairing subject to design changes (in orange).

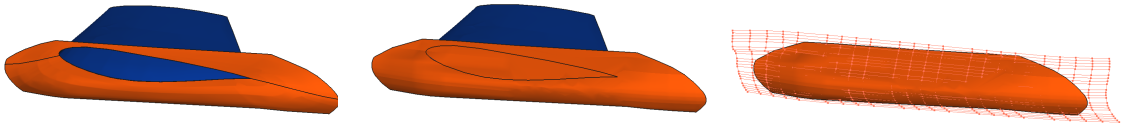

Figure 2. Original trimmed fairing from the inside (left); Untrimmed re-approximated fairing (middle); 22x22 Control Point net of updated fairing (right).

approach, the updated intersection exist in the first instance in terms of updated mesh points rather than as the geometrical entity. In this work, we develop more general, CAD-centric approach and benefit from the functionality provided by the differentiated OCCT. The CAD system recalculates the intersection line, and its geometrical description is then used to propagate mesh changes. Therefore, the role of the CAD system is more profound, which also allows straightforward expansion of methodology to complex cases with multiple and non-trivial intersected patches.

Using the OCCT, we parse the STEP file of the entire NASA CRM and extract the BRep data for the design surfaces. Although only the fairing is defined to move, we consider also the adjacent three wing's patches (Fig. 2) as design areas, since the mesh on these surfaces will be modified. Similarly, we parse the corresponding mesh file and perform point inversions (projections) to determine positions of the surface mesh points on the CAD faces. As a result, each Cartesian (3d) mesh point acquires a pair of 2d parametric coordinates *(u,v)* on the corresponding design patch.

We use OCCT to determine the closed topological bounds of each CAD face: a combination of edges and intersection lines that encircle the face (Fig. 3). On these boundaries, we generate artificial/bogus mesh points (boundary mesh points also have 2d coordinates (u_b, v_b) belonging to the face) and use them to control the inner points mesh movement. Afterwards, any movement of the fairing triggers the following chain of actions for each design face:

(i) by means of OCCT we recalculate all intersections and use them to reconstruct the topological boundaries of the face;

(ii) on each topological boundary (whether it is the intersection or the edge) we update the artificial boundary mesh points $({u_b}^{upd}, {v_b}^{upd})$. We maintain the constant arc-length ratio between each boundary point, based on 1d parametrisation of the corresponding boundary, ensuring uniform distribution in 2d parametric space (Fig. 3 (right));

(iii) we apply the inverse distance weighting[24](IDW) in 2d space of each face to morph the surface mesh points (u, v). We use (u_b, v_b) and $({u_b}^{upd}, {v_b}^{upd})$ as the boundary conditions for the method;

(iv) finally, we use the updated parametric coordinates to calculate 3d Cartesian surface mesh displacements, and propagate them into the volume mesh also using IDW as described in Sec. II.B ;

The algorithm described above has a hierarchical structure when one starts with mesh displacement in 1d space, then these displacements are propagated within 2d parametric space of each surface and only then it is available as an updated 3d Cartesian mesh. Alternatively, the step (iii - smoothing in the parametric space of the face) could be substituted by application of IDW to the inner surface mesh directly in the 3d space. An additional step will be needed here - re-projection of updated 3D points back to the CAD surfaces. The advantage of this approach could be seen for the multi-patch intersection cases, as mesh movement is no longer restricted within each patch. This would allow the surface mesh to "slide" between adjacent patches, avoiding undesirable mesh stretches appearing for the large deformations.

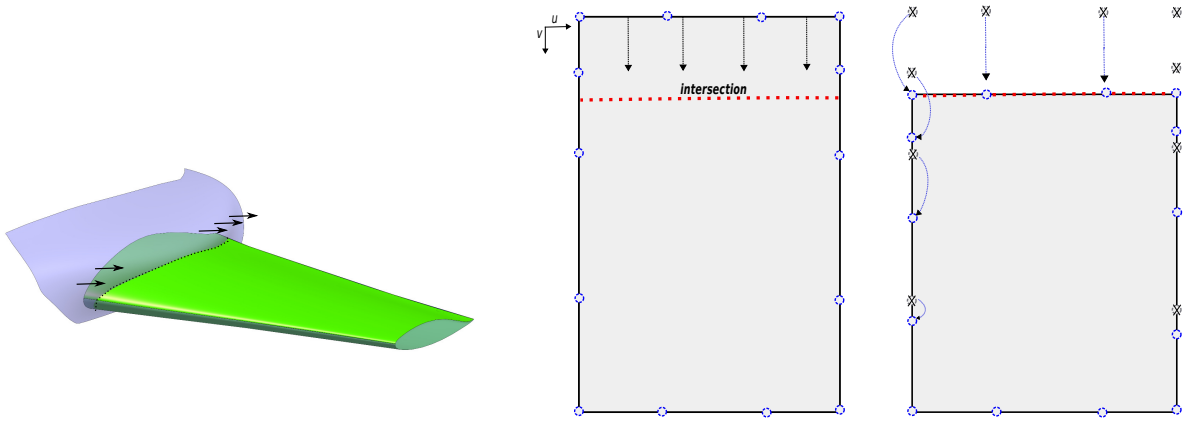

Figure 3. Fairing movement (left); Initial position of the boundary mesh points in 2D parametric space (middle) and the updated positions used then for IDW (right).

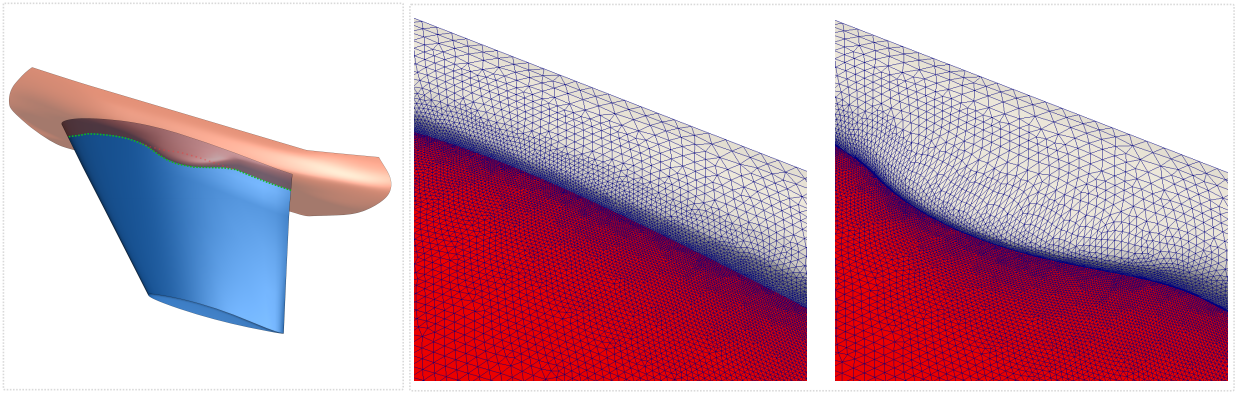

Figure 4. The fairing perturbation (left); The original and perturbed surface meshes (right).

In the Fig. 4 (left), we deliberately perturb several control points in the vicinity of the wing junction, such that the fairing creates a bump on the wing. The results of steps (ii)-(iii) (2d mesh smoothing) are shown for the parametric spaces of the fuselage and one of the wing faces in the Fig. 5 and Fig. 6 respectively. As a consequence of this 2d mesh movement, the corresponding 3d mesh 'follows' the CAD geometry as shown in Fig. 4 (right).

II.H. CAD Sensitivities including Intersections

As we build our framework for the intersections and the corresponding mesh movement algorithm on top of the differentiated OCCT, the complete process (i)-(iv) is also algorithmically differentiated. Therefore, the code is capable of providing exact gradients of the above mentioned algorithm with respect to the fairing control point movements. On the Fig. 7, the magnitudes of the CAD sensitivities with respect to the movement of three different control points in the direction towards the wing are shown. The variation of the first control point (red) corresponds to the finite step deformation showcased in the previous subsection. As a result, the CAD sensitivity resembles the mesh changes on both the fairing and the wing surface (Fig. 4). The second control point (blue) is located further from the junction, and due to the locality of NURBS, does not influence the surface at the wing junction. Therefore, the obtained sensitivity does not differ from the one, which would be obtained in the case without intersections. Finally, the last control point influence the intersection between all three wing surfaces (top, bottom and the trailing edge), hence the non-zero CAD derivatives are present on all of them.

The sensitivity results were compared against the finite differences and showed mutual agreement, thus providing confidence in the differentiated OCCT and algorithms used in this work. Note that finding a proper step size for finite differences is a challenging task therefore making differentiated codes a desirable

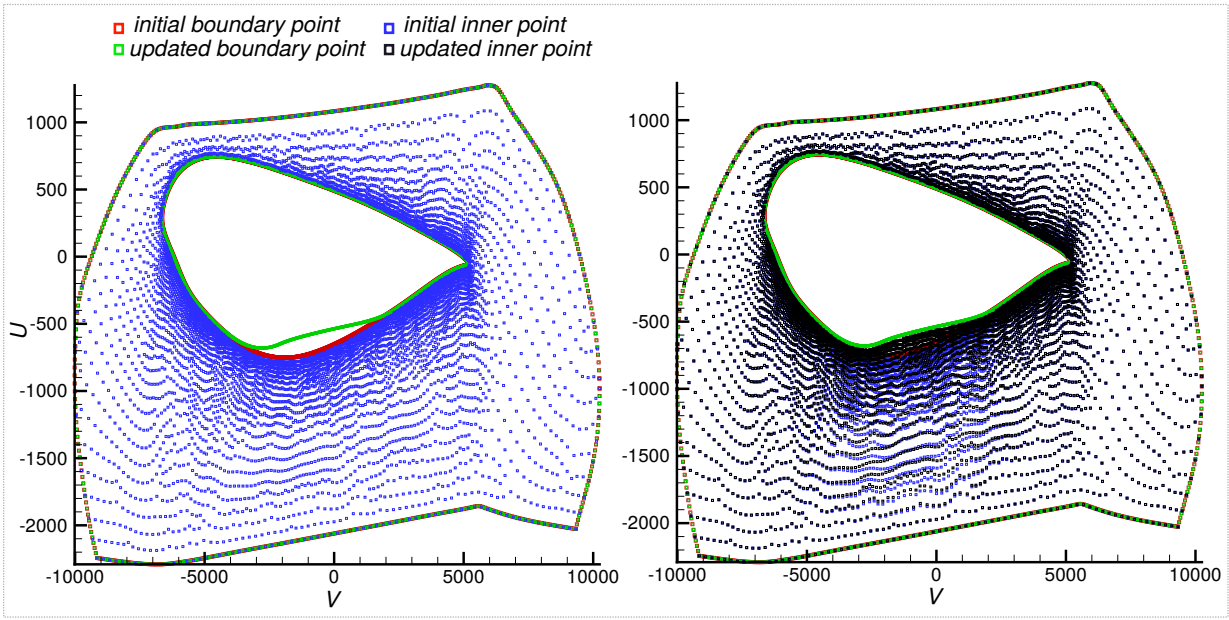

Figure 5. The original (left) and perturbed (right) surface meshes in 2d parametric space of the fuselage.

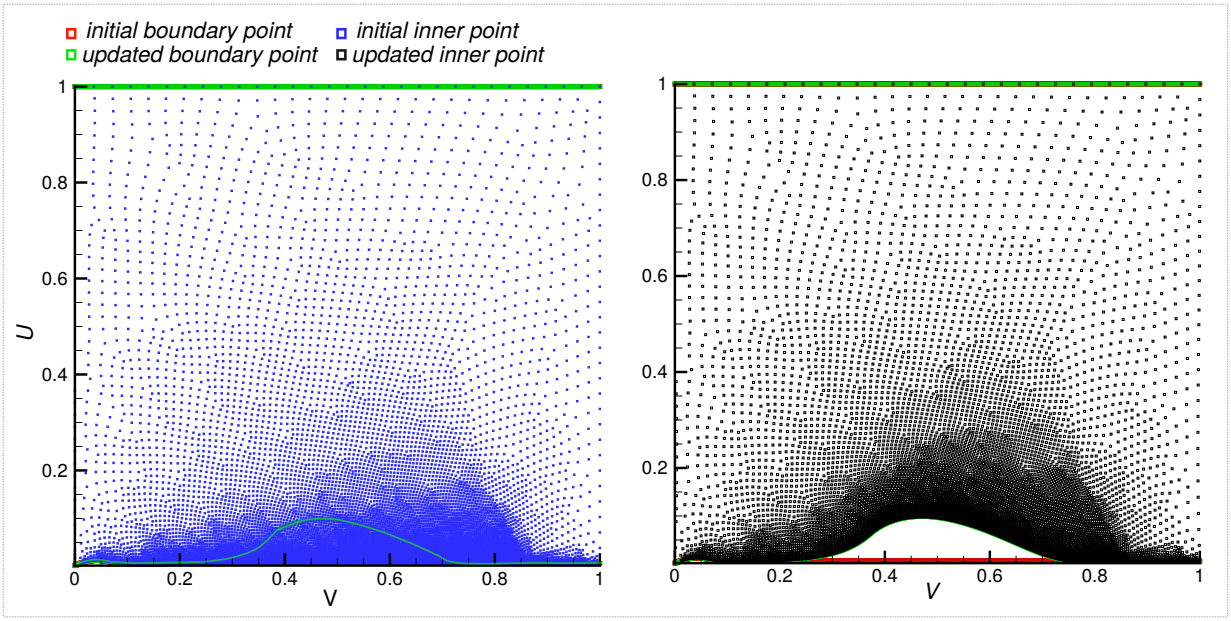

Figure 6. The original (left) and perturbed (right) surface meshes in 2d parametric space of the bottom wing's surface.

feature for CAD-based methods. We divide our final 507 design variables into three blocks and apply vector differentiation consequently. The time spent on this is lower than for the corresponding low-fidelity CFD calculation.

III. CRM Optimisation Results

The NASA CRM model was created for the fourth AIAA CFD Drag Prediction Workshop[25] to assess the state-of-the-art computational methods as practical aerodynamic tools for force and moment prediction

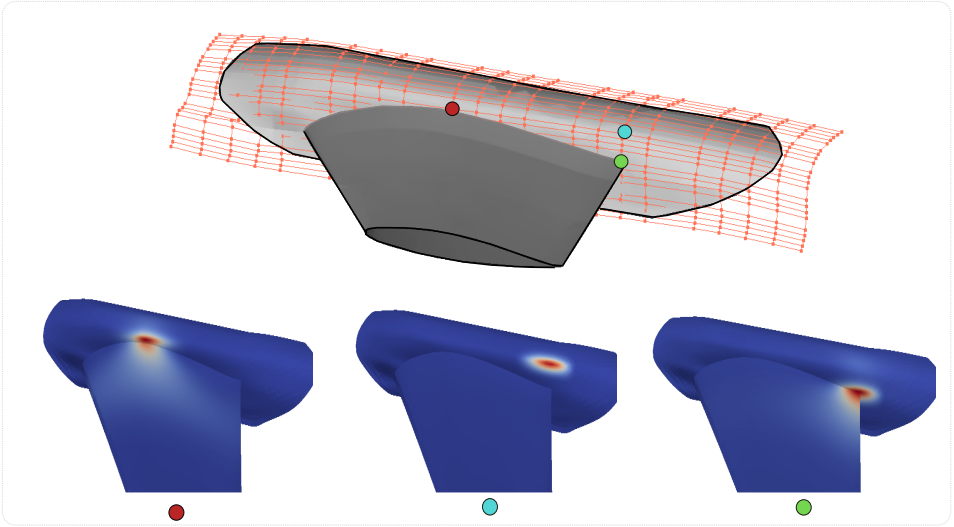

Figure 7. Magnitude of CAD Sensitivities w.r.t. three different control points.

on increasingly complex airframe geometries. Nacelle and pylon are not included here for simplicity and we use wing-body configuration. The CAD geometry and comprehensive case description can be found on the workshop's on-line resources. Calculations are performed at defined Mach number $Ma = 0.85$ and a 2.5° angle of attack.

As we are focusing on demonstration of a complex CAD-based method with completely differentiated design chain, at this point we have considered a rather simplistic flow conditions and optimisation goal. The CFD mesh produced using the provided CAD geometry has 1.2 million tetrahedral elements with 200k grid points. The Euler equations are solved. The objective function is the drag on the design surfaces (fairing and the adjacent wing patches) and no constraints are imposed. At each design step, the steady state flow and adjoint solution are first computed, the surface sensitivity is assembled, and provided to the steepest descent optimiser which returns the step size of the next perturbation. The general goal of the fairing optimisation, as shown in authors' work on DLR F6 case, is to suppress the separation at the wing-fairing junction. Due to the simplified CFD, this phenomena is not present for the CRM case. In the inviscid case, the drag is caused by the presence of a shock at the transonic flight condition.

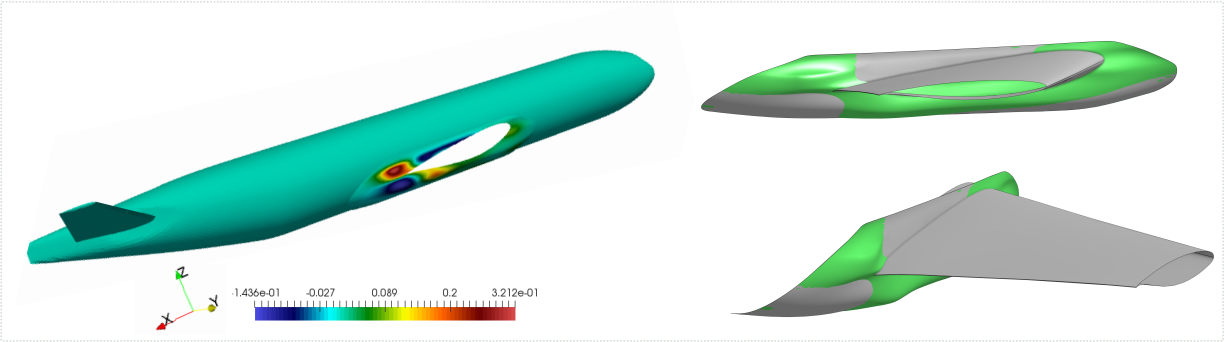

Figure 8. Deformation magnitude on the initial geometry (left); Initial and updated (green) design surfaces (right).

After 10 optimisation iterations, the drag is reduced by 7%. In Fig. 9, it can be seen that due to a Mach number redistribution along the chord, the shock position is changed, leading to a smaller lift. The reduction in the total drag probably originates from a smaller induced drag. The main reason this is the case is that the current optimisation is based on the Euler equations and no viscous effect are considered.

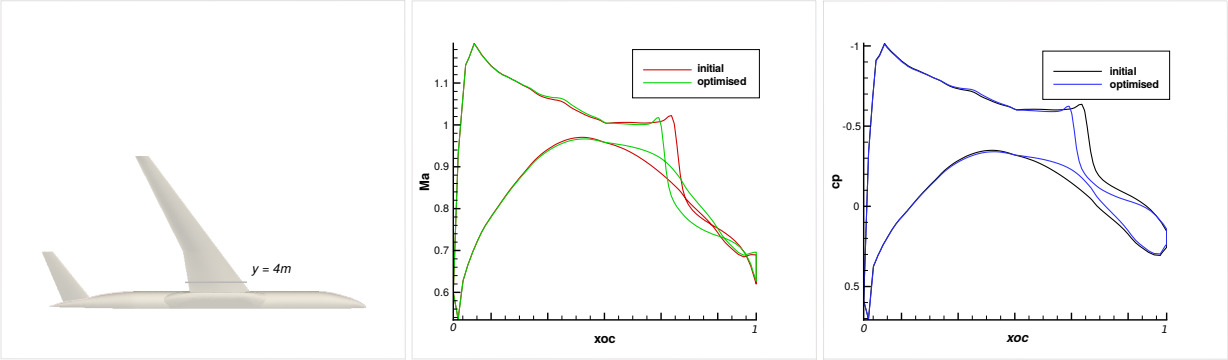

Figure 9. Optimised results

IV. Summary

We have successfully demonstrated the integration of a large-scale CAD system into the design chain for shape optimisation of an aircraft. The approach allows to maintain CAD-models throughout the optimisation loop which is an enabling factor for the multi-disciplinary environment. The OCCT CAD kernel that is algorithmically differentiated was extended to provide sensitivities for the parametrisations with multiple intersecting patches. These gradients are obtained efficiently, accurately and robustly. Together with differentiated adjoint solver and mesh movement techniques we have assembled a fully differentiated design chain.

The simplified flow conditions that were used to drive the drag optimisation of the NASA CRM allowed to showcase the strength and potential of the CAD-based method. The further investigation of the model will be considered under turbulent flow condition, where the fairing intersection will play a more significant role in the overall aerodynamic efficiency.

Acknowledgements

The research leading to these results was co-funded by:

- IODA project - Industrial Optimal Design using Adjoint CFD. IODA is *Marie Sklodowska-Curie Innovative Training Network* funded by European Commission under Grant Agreement No. 642959.

- Innovate UK, the UK's innovation agency, as part of the Enhanced Fidelity Transonic Wing project.

References

[1]Stephan Schmidt, Caslav Ilic, Nicolas Gauger, and Volker Schulz. Shape gradients and their smoothness for practical aerodynamic design optimization. DFG SFB 1253 Preprint-Number SPP1253-10-03 http://, Universität Erlangen, April 2008.

[2]A. Jaworski, J.-D. Müller, and J. Rokicki. One-shot optimisation with grid adaptation using adjoint sensitivities. In *Proceedings ECCOMAS 2012*, pages 3685–3693, 2012.

[3]A. Jameson and A. Vassberg. Studies of alternative numerical optimization methods applied to the brachistochrone problem. *Computational Fluid Dynamics Journal*, 9(3), 2000.

[4]R. Jesudasan, X. Zhang, M. Gugala, and J.-D. Mueller. CAD-free vs CAD-based parametrisation method in adjoint-based aerodynamic shape optimization. In *Proceedings ECCOMAS 2016*, 2016.

[5]Shenren Xu, Wolfram Jahn, and Jens-Dominik Müller. CAD-based shape optimisation with CFD using a discrete adjoint. *International Journal for Numerical Methods in Fluids*, 74(3):153–68, 2013.

[6]T.T. Robinson, C.G. Armstrong, H.S. Chua, C. Othmer, and T. Grahs. Optimizing parameterized CAD geometries using sensitivities based on adjoint functions. *Computer-Aided Design & Applications*, 9(3):253–268, 2012.

[7]J. F. Dannenhoffer III and R. Haimes. Design sensitivity calculations directly on CAD-based geometry. In *53rd AIAA Aerospace Sciences Meeting*, page 1370, 2015.

[8]S. Xu, S. Timme, O. Mykhsakiv, and J.-D. Mueller. Multi-point adjoint-based shape optimization for wing-root fairing design. *Submitted to Aerospace Science and Technology*, 2017.

[9]S. Auriemma, M. Banovic, O. Mykhaskiv, H. Legrand, J.-D. Müller, and A. Walther. Optimisation of a U-bend using CAD-based adjoint method with differentiated CAD kernel. In *ECCOMAS Congress*, 2016.

[10]M. Banovic, O. Mykhaskiv, S. Auriemma, A. Walther, H. Legrand, and J. D. Müller". Automatic differentiation of the Open CASCADE Technology CAD system and its coupling with an adjoint cfd solver. In *Optimization Methods and Software*, 2017.

[11]O. Mykhaskiv, M. Banovic, S. Auriemma, A. Walther, and J.-D. Mueller. NURBS-based and Parametric-based shape optimisation with differentiated CAD kernel. *Submitted to Computer-Aided Design & Applications*, 2017.

[12]Zhoujie Lyu, Gaetan KW Kenway, and Joaquim R.R.A. Martins. Aerodynamic shape optimization investigations of the common research model wing benchmark. *AIAA Journal*, 53(4):968–985, 2014.

[13]F. Palacios, T. D. Economon, and J. J. Alonso. Large-scale aircraft design using su2. In *53rd AIAA Aerospace Sciences Meeting*, page 1946, 2015.

[14]Song Chen, Zhoujie Lyu, Gaetan KW Kenway, and Joaquim RRA Martins. Aerodynamic shape optimization of common research model wing–body–tail configuration. *Journal of Aircraft*, 53(1):276–293, 2015.

[15]L. Piegl and W. Tiller. *The NURBS book*. Springer Science & Business Media, 2012.

[16]A. Jameson. Aerodynamic design via control theory. *Journal of Scientific Computing*, 3:233–260, 1988.

[17]Shenren Xu. *CAD-based CFD shape optimisation using discrete adjoint solvers*. PhD thesis, Queen Mary University of London, 2015.

[18]S. Xu, D. Radford, M. Meyer, and J.-D. Müller. Stabilisation of discrete steady adjoint solvers. *Journal of Computational Physics*, 299:175–195, 2015.

[19]L. Hascoët and V. Pascual. Tapenade 2.1 user's guide. Technical Report 0300, INRIA, 2004.

[20]M.B. Giles. On the use of runge-kutta time-marching and multigrid for the solution of steady adjoint equations. NA Group Report NA-00/10, Oxford University Computing Laboratory, 2000.

[21]A. Walther and A. Griewank. Getting started with ADOL-C. *Combinatorial scientific computing*, 2012.

[22]M. Burgos and M. Jaime. Adaptation of the computational grid to a moving wing-fusselage intersection via nurbs and radial basis. *6th European Conference on Computational Fluid Dynamics*, 2014.

[23]M. Burgos and M. Jaime. *NURBS-Based Geometry Parameterization for Aerodynamic Shape Optimization*. PhD thesis, Telecomunicacion, 2015.

[24]J. A. Witteveen. Explicit and robust inverse distance weighting mesh deformation for CFD. In *48th AIAA Aerospace Sciences Meeting*, volume 165, 2010.

[25]J. Vassberg, M. Dehaan, M. Rivers, and R. Wahls. Development of a common research model for applied cfd validation studies. In *26th AIAA Applied Aerodynamics Conference*, page 6919, 2008.