# Integration of CFD in Computational Design

## *An evaluation of the current state of the art*

*Angelos Chronis[1], Alexandre Dubor[2], Edouard Cabay[3], Mostapha Sadeghipour Roudsari[4]*
*[1,2,3]Institute for Advanced Architecture of Catalonia [4]University of Pennsylvania*
*[1,2,3]{angelos.chronis|alex|edouard.cabay}@iaac.net [4]mostapha@ladybug.tools*

*The integration of building performance feedback in the design process is increasingly considered as a key aspect of the decision support framework that drives current high performance architecture, from early conception to fabrication. Although on other aspects of building performance there has been significant recent development on BPS integration in computational design, the integration of CFD is still largely unexplored, despite its significance in numerous design problems. This paper reviews the current state of advancement of integrated CFD simulation tools in computational design frameworks by evaluating three different integration approaches, each representing a different level of integration of CFD solvers within the commonly used computational design frameworks today. The objective of the study is neither to provide an extensive evaluation of all available CFD frameworks nor to assess the specific performance of the problem at hand, but rather to evaluate the potential and limitations of each integration approach from the perspective of the computational design user.*

**Keywords:** *Computational Fluid Dynamics, Simulation, Integration, Computational Design*

## INTRODUCTION

The integration of building performance feedback in the design process is increasingly considered as a key aspect of the decision support framework that drives current high performance architecture, from early conception to fabrication. The computational paradigm shift that leads the research in design methodology has been closely followed by an intensive development and integration of building performance simulation (BPS) tools that aim to support the performance goals of novel form-found and optimized architectures.

Despite this increasing significance of BPS tools and their consequent continuous development and integration in CAAD software, it is generally acknowledged (Clarke and Hensen 2015, Weytjens et al 2012) that currently available BPS tools are substantially deficient in providing optimized or even adequately in-

formed design solutions. Their deficiencies relate not only to interoperability, modeling resolution, computational efficiency and lack of domain knowledge but also to the uncertainty of modern complex design problems (Hopfe and Hensen 2011, Hanna et al 2010). Furthermore, in the light of the evolving incorporation of computational design methodologies in mainstream architectural practice and subsequent research focus on more advanced, adaptive and real-time performing architectural solutions this complexity is inevitably due to increase. It follows that for these design spaces of higher complexity to achieve an adequately higher level of performance goals, a higher level of integration of BPS tools is needed, not only in mainstream CAD software but more importantly within the computational design frameworks that shape the forefront of performance-driven design processes.

This study contributes to the evaluation of BPS tools that are currently integrated in computational design, focusing on air-flow performance and consequently the integration of computational fluid dynamics (CFD) simulations. Although on other aspects of building performance, such as daylight, solar radiation and energy consumption, there has been significant recent development on BPS integration in computational design, the integration of CFD is still largely unexplored, despite its significance in numerous design problems, such as natural and cross ventilation, heating ventilation and air conditioning (HVAC), energy consumption, pedestrian comfort, structural performance and many more. Considering the performative potential that can be achieved through the geometrical complexity of the forthcoming additive manufacturing paradigm, this integration is a key step in enabling design solutions that will harness this potential.

The study aims to assess the current state of integration of CFD simulations in computational design through the case study of optimizing a 3D printed clay wall in terms of its natural ventilation potential. The scope of the study focuses purely on the potential of different simulation approaches that are currently available to architects and not on the design of the 3D printed wall per se. The objective of this approach is to evaluate both the accessibility and user-friendliness of available tools as well as their potential use in computational design and optimization processes.

## PROBLEM DEFINITION

Three different approaches were undertaken and evaluated, within this scope, that represent different levels of CFD integration within the computational design framework. Namely, a CAAD-CFD approach using the RhinoCFD plugin for McNeel Rhinoceros, a CFD integrated computational design approach using the Butterfly add-on that integrates OpenFOAM into the Grasshopper plugin for Rhinoceros as well as a fully integrated CFD approach using a Fast Fluid Dynamics solver developed in the Processing programming language (Chronis et al 2011).

### *The Case Study of a 3D Printed Clay Wall*

The case study undertaken is a part of an overarching study on the use of uncured clay 3D printing for the fabrication of a performative wall. The overarching study is the outcome of the Open Thesis Fabrication program, at the Institute for Advanced Architecture of Catalonia which is a research program within which a group of researchers are exploring the potential of digital fabrication technologies and in this case, additive manufacturing with uncured clay. The overall research objective of the study is to use the thermodynamic properties of clay and the potential of robotic 3D printing to create novel performative uses of an ancient building material. The development of the project involved thorough research of the climatic phenomena and the material behavior, physical tests using performance monitoring machines, such as a hygrothermal monitoring apparatus and a load bearing machine and digital simulations using solar radiation, daylight, thermal conductivity, thermal convection, thermal mass and structural analyses. The final outcome of the research program has been the design and fabrication of a real scale per-

formative wall prototype that has been erected in the campus of the school (see Figure 1)..

Under the scope of this overarching study the objective of the part presented here is to optimize the wall's openings to achieve an optimum ventilation strategy for the wall. Taking advantage of the Bernoulli principle, the objective of the study is to optimize the inner wall geometry to achieve maximum cooling of the wall's compartments during the summer period. As stated above, this study does not focus on the design problems of the 3D printed wall or the overarching research questions of fabrication, performance and evaluation of the wall design itself, but rather on the integration and the usability of CFD simulations in the computational design framework that is used for the design of the wall.

The potential geometrical complexity that is allowed by the additive fabrication method opens a large solution space for experimentation which on one hand allows for greater performance gains but on the other makes insight of the performance of this solution space more tedious. It follows that a computational design approach is inevitable to explore this solution space and identify performance trends and consequent design strategies. However, the integration of a valuable performance feedback mechanism in terms of the ventilation performance of the wall has been proven problematic, hence the development of this study aims to tackle this integration and propose new approaches to make this integration more feasible.

### *Scope and Limitations*

Although the scope of this study is to assess the current state of the art of the integration of CFD simulations in computational design frameworks, the three different approaches taken are only a subset of the integrated CFD frameworks available today. This subset is however a good representation of the different approaches available for an architect or building engineer today. The selection of the tools that are used here was done based on certain criteria,

Figure 1
Final Full Scale Prototype of the 3D Printed Wall

such as the accessibility in terms of cost and expertise of the tools, the breadth of the user base of available tools and the integration of the CFD simulation frameworks with mainstream CAAD and computational design frameworks, such as the cases of Grasshopper and Processing. A different approach that has been considered, for example, is the mainstream CFD engineering package, ANSYS Fluent and its built-in optimization routines. The prohibiting cost of the license of the package though as well as the lack of its integration with a CAAD package have been fundamental in omitting it from this review. Other approaches, such as Autodesk Flow and its plugin for Revit, Solidworks Fluid Flow or RealFlow have also been deliberately omitted from this review, as they are either equivalent to the approaches taken here or, similarly to ANSYS, not integrated within the architecture-related design frameworks.

It should also be made clear that the aim of the comparison between the three different approaches is not an optimum integration method or the achievement of an optimum performance for the case study, but merely the assessment of the available tools and frameworks and their potential and limitations. Each of the different approaches achieves a different level of accuracy in the simulation, a different level of computational design integration and a different level of usability and accessibility for the architect, thus providing a range of levels for these objectives, each with its own merits depending on the problem at hand. Important issues such as domain knowledge of the users, open-source and free availability or documentation and community support are also important drivers in this comparison and any future use of it.

## EVALUATION OF CFD INTEGRATION AND OBJECTIVES

It is beyond the scope of this paper to describe the process of a CFD simulation workflow, however it is important to define the criteria upon which the integration is evaluated within the scope of this study. The first and most important aspect of this integration is the interface between the architectural and the simulation geometrical representations. As the focus of the integration is the potential of the CFD simulations within a computational design framework, a seamless link between the architectural geometry and the simulation model is essential. This first step involves the automated set-up of the domain, the meshing and the boundary conditions. Issues such as the adaptability of the mesh, the level of resolution and refinement and domain fitting to the architectural geometry are significant aspects of a seamless integration. The geometry of the 3d printed wall has been designed parametrically with Grasshopper for Rhino, which has served as the main platform for integration with all other simulation and analyses tools of the overarching study. Therefore, the geometry pipeline is assessed in relation to the main design platform, Rhino and Grasshopper. (see Figures 2 and 3)

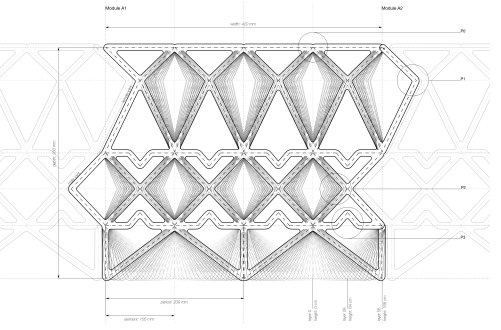
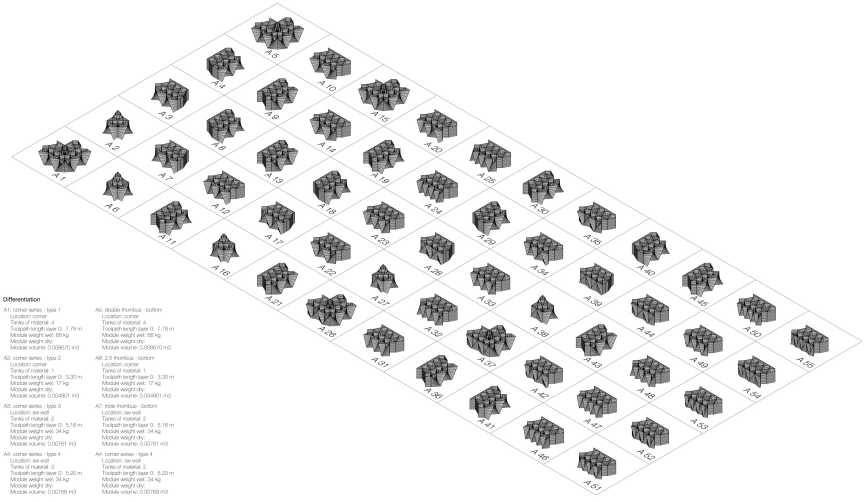



Figure 2
Parametrically Defined Module Infill Logic

The most important part of the simulation is of course the solver itself, therefore the speed, accuracy and stability of the solver are also crucial. Again, as the focus here is on the integration and not on the performance metrics of either the solver or the problem at hand, validations of the results are not performed and the accuracy of the solvers is derived from either their established status (PHOENICS and OpenFOAM) or based on previous research (FFD) and is considered as such. Furthermore, the accuracy of

Figure 3
Parametrically Derived Catalog of Parts for Fabrication



each solver assessed is considered along with the potential scope of their integration and not as a means of comparison between the solvers.

### *Optimization and Iterative Solving*

Finally, an important aspect of the CFD integration is the post-processing of results, both for communicating the analysis to the user, but also for enabling iterative solving and thus integration of optimization routines within a computational design process. This latter objective of iterative solving and optimization, is key in harnessing the potential of integrating CFD in the computational design workflow, and the different approaches discussed here are achieving it at different levels, if at all. The aforementioned objectives of seamless integration, accessibility and user-friendliness are all prerequisites to this goal but, furthermore, to achieve a performance feedback mechanism for our overly complex computational design problems there is a need for dynamic representation simulation models. These entail advanced simulation concepts such as dynamic meshing, multi-physics solvers, dynamic resolution domains, customizable routines and high-performance computing, which although not discussed in detail here, are all considered as potential further steps on each evaluated approach.

## THREE INTEGRATION APPROACHES

### *RhinoCFD*

The first approach for CFD integration was done using the RhinoCFD plug-in for Rhinoceros (Rhino). RhinoCFD is a very recently developed integration framework for CHAM's PHOENICS and its special purpose variant, FLAIR which focuses on building air flow and thermal simulation. PHOENICS is well established in building physics research and practice and it has been a popular software solution among educators, engineers and integrated architectural practices. The integration of PHOENICS and FLAIR within the Rhino CAAD framework is in its infantry, nevertheless it has been very well received as a first integration step from the Rhino users' community. PHOENICS runs externally to Rhino, therefore the plug-in serves as an integration tool that, based on the native Rhino geometry, creates the mesh and simula-

tion domain and model and then uses the PHOENICS solver to run the case externally. The results can be visualized within Rhino as well as exported and visualized in other platforms. Although RhinoCFD is fully capable of running advanced simulation scenarios, not all of PHOENICS' capabilities are exposed to RhinoCFD yet, which can be limiting its potential for use in a computational design framework. (see Figure 4)

RhinoCFD' s main benefit is its integration within Rhino and its native support of Rhino's geometry. The meshing and domain modeling are all done within Rhino and they are very straightforward. RhinoCFD uses an orthogonal mesh, which is adapted to different regions by detecting the geometry of the model. The regional control of the mesh is again straightforward although not necessarily very efficient in models with a large number of objects. The setting up of the domain is equally straightforward and with a few steps the user can quickly set up a basic wind-tunnel simulation, even with limited simulation expertise and no previous experience in CFD. The solver, which as mentioned runs externally and can run in parallel in the latest version of the plug-in, therefore the speed of the simulation is relatively fast. Results can be probed and visualized within the same file in the same straightforward way as well as exported and used to query any part of the domain, in order to progress a - manual - optimization process.

From a user perspective, RhinoCFD has a very quick learning curve and is very accessible to non-experts. In our case, the researchers were trained to use the software and could very quickly create and run simulation cases which helped them understand the effect of the openings of the wall in the internal channeling of the air. PHOENICS FLAIR, which is the core of the solver underneath RhinoCFD also has great potential in terms of its use in environmental engineering and, specifically in our case, its energy and radiation models are essential in progressing the study, as the thermodynamic effects of the

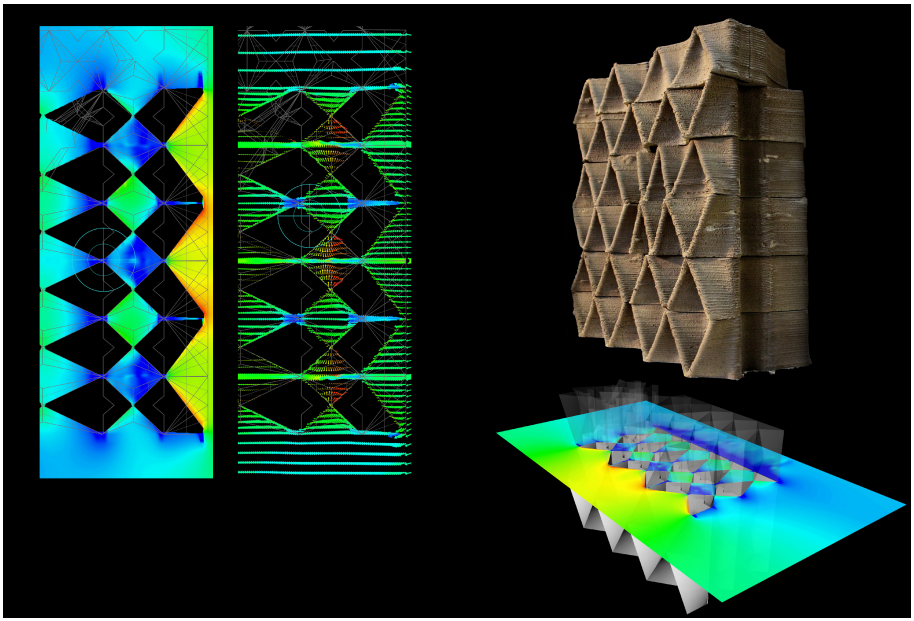


Figure 4
RhinoCFD Simulations of the 3D Printed Wall

wall depend on them. The modeling and meshing parts using this CAAD-CFD integrated approach are quite straightforward. RhinoCFD uses an orthogonal mesh, which is on one hand easy to set up and manipulate but on the other not as efficient for complex geometries. A body-fitted mesh is available in the full version of PHOENICS, but not exposed to RhinoCFD.

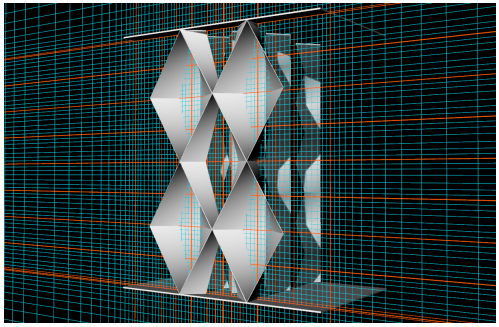
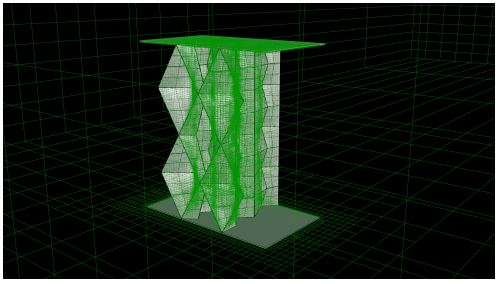


Figure 5
RhinoCFD Orthogonal, Non Body-Fitted Mesh

Figure 6
Meshing of the Model with Butterfly add-on in Grasshoper

The most important limitation of RhinoCFD is that at its current stage it does not allow an iterative optimization routine to be based on it, as it is designed for single runs. PHOENICS has optimization routines which are though both limited and not yet implemented within RhinoCFD. A potential solution to this problem would be the integration of RhinoCFD within Grasshopper, a step that is currently being tested. Exposure of RhinoCFD to Grasshopper could allow the programmatic run of the solver from a Grasshopper component, thus allowing an iterative simulation, coupled with the many optimization capabilities of Grasshopper. This would still be a limited approach as the simulation domain needs to be revoked every time due to the lack of a dynamic domain and mesh but it would nevertheless allow for a valuable CFD integration in computational design. The solver is also parallelizable, something that could help more in enabling iterative optimization routines. (see Figure 5)

Overall, RhinoCFD is a great integration effort for CFD in CAAD. Its user friendliness and seamless integration make it a great introductory CFD simulation platform for architects with limited domain knowledge and its robustness, scalability and engineering capability give it great potential for more complex use cases. In our study, we have used RhinoCFD extensively in a trial and error optimization process and therefore we consider this approach as a very valuable first step for integration. As mentioned, exposure of the solver to Grasshopper would allow coupling with iterative optimization processes, thus this is considered as a next step for our study. On the downside, RhinoCFD is powered by PHOENICS, which although reasonably priced in comparison to other CFD platforms of the AEC it is nevertheless, proprietary software, thus its development and source are depending on CHAM.



### *Butterfly*

The second approach undertaken is the only currently available CFD integration for the Grasshopper computational design platform. The Butterfly add-on is connecting the most recognized open source CFD solver, OpenFOAM to the Grasshopper plugin and to Dynamo for Autodesk Revit. OpenFOAM is developed for Linux-based operating systems and thus it does not run natively on Windows, the main operating system supported by Rhino and most other CAAD software. It is however distributed for Windows encapsulated within a Docker container, which runs a Virtual Machine (VM) that runs Linux. The complexity of setting up the VM that runs OpenFOAM is currently the main limitation of this approach and the main reason that it has not been our main method of simulation in this study. Despite its difficulty to set up

for Windows though, OpenFOAM is a very powerful and robust CFD solver that has endless capabilities. Butterfly is also relatively new as an integration effort and thus it exposes a small subset of OpenFOAM' s capabilities to Grasshopper, which is though not a limitation at this stage. The most important benefit of this integration is that both Butterfly and OpenFOAM are open-source and freely available and supported by large communities of users and developers who are constantly extending the capabilities of the solver and its integration, each evidently at a different scale. They are also both well documented, although Butterfly does require some background knowledge on CFD modeling and simulation to properly set up and run

Butterfly runs completely integrated within Grasshopper, although most of the preprocessing, the simulation and part of the postprocessing is happening externally. Butterfly is essentially wrapping all OpenFOAM commands and necessary files in a number of Grasshopper components that communicate the commands to the VM and write these files to a shared drive location. A great effort has been done to simplify this process as much as possible and provide default configurations whenever needed so that novice users can easily set up and run simulations, but due to the more advanced nature of the solver, the domain set up is not always straightforward. OpenFOAM' s meshing process runs in two steps, creating first a coarse mesh and then a finer, body-fitted one. There are many capabilities for mesh refinement and optimization which are exposed by Butterfly and many more that are provided by OpenFOAM. As the geometry is referenced directly from Grasshopper, the whole process can be parametrically controlled at every step. The integrated part of OpenFOAM currently allows for incompressible flow and heat transfer but again, there are endless possibilities for further development, based on the breadth and depth of OpenFOAM' s capabilities. The solver can also run parallelized, depending on the resources exposed to the virtual machine. The results of the simulation are probed externally but can be visualized and real-time queried. One significant aspect of the solver is that the simulation parameters can be altered while the simulation is running, allowing even more possibilities for optimization and control. (see Figure 6)

The open architecture of the solver, in combination with the full and open integration of the Butterfly add-on in Grasshopper make it an excellent base for coupling CFD with optimization and computational design. The geometry, domain, meshing and all other simulation parameters are open for customization, automation and iteration giving endless possibilities for CFD-informed performative design. The only limitation for iterative optimization methods is Grasshopper's linear architecture. On the other hand, the main limitation of the approach of Butterfly is, as mentioned, the very cumbersome process of setting up the OpenFOAM, which is the reason why our experimentation with it has been minimal during the initial parts of this study. The complexity process is related not only to the VM architecture but also to basic operating system limitations, such as administrative rights or differences in Windows versions which have been proven very difficult to overcome in a group of people working with different system setups. These issues could potentially be tackled in the future with batch setup processes, although this is also not straightforward to implement. Further to the software architecture issues, OpenFOAM and consequently Butterfly are considerably more complex in usage and not as accessible to novice users. OpenFOAM' s documentation is broad but not focused to non-experts. Overall, the potential of this approach is considered the most promising for the purpose of the study, although it has not been fully explored yet. At this initial experimentation, an iterative optimization routine has been performed as a proof of concept and the aim is to further study the limitations of the approach by involving more complex geometries and simulation problems. However, at its current stage, the approach has not been useful for the problem at hand, as it has been very difficult to set up both the solver and at a later stage the complex

geometry and domain of the wall optimization problem.

### *Processing FFD*

The third approach taken is using a self-contained and fully integrated CFD solver with some built-in geometry optimization capabilities, developed in the Processing programming language. Previous studies (Chronis et al 2011, Athanailidi et al 2014) have demonstrated the potential of integrating a less computationally expensive CFD simulation engine not only in early design stages, to inform preliminary design decisions, but furthermore in form finding optimization processes using genetic algorithms. The Fast Fluid Dynamics (FFD) solver has been developed based on a numerical scheme initially developed for the computer graphics industry (Stam 1999) and it has been previously validated for limited design problems (Zuo and Chen 2007). The integration of the solver in Processing as well as the geometry meshing, results probing and visualization and optimization routines are all developed by one of the authors and thus there is full control over the entire preprocessing, simulation and postprocessing workflow.

The full integration of the solver code within problem-specific developed Processing applications has been proven to provide a great potential for iterating solving and customization in terms of the simulation and optimization processes but it has been very poor in terms of its geometrical capabilities as well as the performance and validity of results, due mainly to the limited accuracy of the solver as well as the lack of resolution and mesh refinement. The developed application uses an orthogonal square mesh, with no refinement regions and very limited boundary condition options, which makes it very hard to define a complex problem adequately. Moreover, the solver lacks proper turbulence modeling and although it has a basic buoyancy model, it is very inaccurate and thus not suitable for thermal modeling of the problem. Considering that the simulation part can be replaced in the future by a more accurate and capable CFD solver, either in the form of a library or another natively coded solver, the evaluation of the approach focuses on the potential of a fully integrated CFD simulation within a programming language that the user controls and not on the results of the solver.

One of the most important limitations of the approach, at its current stage, is that it requires a custom-made application for every given design problem, as the geometrical representation of the problem is hard coded within the application. To overcome this problem, after our initial experimentation, we integrated the solver in an experimental plugin for Grasshopper with an aim to provide full computational design flexibility of the solver by incorporating it within the Grasshopper plugin. Therefore, at its current stage, the solver runs externally, but the geometry and simulation parameters are streamed real time from Grasshopper. This integration would allow the user to design the solution space with ease, thus rendering the need for the development of a custom application per case unnecessary. (see Figure 7)

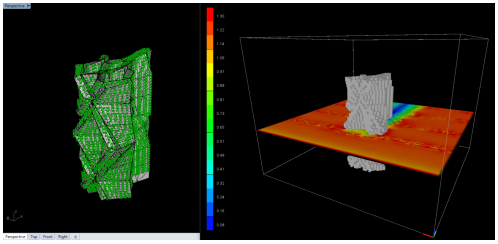


Figure 7
Grasshopper - FFD integration running in Real Time

Although the approach has in principle great potential as it allows full control by the user of the whole simulation engine, including the solver itself, and at a low level, it has generally considered as not appropriate at this stage due to the many problems of the solver. The inaccuracy of the solver, the meshing and flow modeling limitations and the lack of a parametric geometry module within Processing makes it very difficult to work with complex simulation problems. From a user perspective, this approach is also the most inaccessible as it requires programming

knowledge as well as knowledge of the framework itself, thus making it impossible to distribute to a group of researchers, as was the case here. Despite all these limitations though, the potential of using a CFD solver completely incorporated in a computational design framework demonstrates significant customization potential. A further step, for example, could be the incorporation of a GPU based CFD solver with some thermal modeling capabilities, similar to the one used here, within Grasshopper, something that is part of the future work of this study.

## CONCLUSIONS

This study reviews the current state of advancement of integrated CFD simulation tools in computational design frameworks by evaluating three different integration approaches, each representing a different level of integration of CFD solvers within the commonly used computational design frameworks today. The objective of the study has not been to either provide an extensive evaluation of all available CFD frameworks or to assess the specific performance of the problem at hand, but rather to evaluate the potential and limitations of each integration approach from the perspective of the computational design user.

The study demonstrates the limited capabilities of all three approaches in achieving a seamless integration of CFD in computational design and supports the need for further development of such interfaces. Driven by this conclusion, our aim is to continue the development of an integrated CFD solver that would allow designers to get valuable performance insight on air-flow related problems within the computational design framework that they already use.

## ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 642877. The case study is part of the Open Thesis Fabrication program of the Institute of Advanced Architecture of Catalonia with researchers Raaghav Chenthur Naagendran, Sameera Chukkapalli, Iason Giraud, Abdullah Ibrahim, Lidia Ratoi, Lili Tayefi, and Tanuj Thomas supervised by Edouard Cabay and Alexandre Dubor

## REFERENCES

Athanailidi, P, gen Schieck, AF, Tenu, V and Chronis, A 2014 'Tensegrity systems acting as windbreaks: form finding and fast fluid dynamics analysis to address wind funnel effect', *Proceedings of the Symposium on Simulation for Architecture & Urban Design*

Attia, S, Hensen, JL, Beltr?n, L and De Herde, A 2012, 'Selection criteria for building performance simulation tools: contrasting architects' and engineers' needs', *Journal of Building Performance Simulation*, 5(3), pp. 155-169

Chronis, A, Turner, A and Tsigkari, M 2011 'Generative fluid dynamics: integration of fast fluid dynamics and genetic algorithms for wind loading optimization of a free form surface', *Proceedings of the 2011 Symposium on Simulation for Architecture and Urban Design*, pp. 29-36

Clarke, J and Hensen, J 2015, 'Integrated building performance simulation: Progress, prospects and requirements', *Building and Environment*, 91, pp. 294-306

Hanna, S, Hesselgren, L, Gonzalez, V and Vargas, I 2010 'Beyond simulation', *Proceedings of the 2010 Spring Simulation Multiconference on - SpringSim*, New York, New York, USA

Hopfe, CJ and Hensen, JL 2011, 'Uncertainty analysis in building performance simulation for design support', *Energy and Buildings*, 43(10), pp. 2798-2805

Stam, J 1999 'Stable fluids', *Proceedings of the 26th annual conference on Computer graphics and interactive techniques - SIGGRAPH*, New York, New York, USA, pp. 121-128

Weytjens, L, Attia, S, Verbeeck, G and De Herde, A 2011, 'The 'Architect-friendliness' Of Six Building Performance Simulation Tools: A Comparative Study', *International Journal of Sustainable Building Technology and Urban Development*, 2(3), pp. 237-244

Zuo, W and Chen, QY 2007 'Validation of fast fluid dynamics for room airflow', *10th International IBPSA Conference (Building Simulation 2007)*