12-2016

Affordance-based Design Product Evolution Using Customer Feedback

Ivan Mata
Clemson University, imata@g.clemson.edu

AFFORDANCE-BASED DESIGN PRODUCT EVOLUTION USING CUSTOMER FEEDBACK

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Mechanical Engineering

---

by
Ivan Mata
November 2016

Accepted by:
Dr. Georges Fadel, Committee Chair
Dr. Mary Elizabeth Kurz
Dr. Joshua Summers
Dr. Gregory Mocko
Distinguished External Reader: Dr. Kemper Lewis, SUNY at Buffalo, NY

Abstract

Designers can benefit from involving the user in the product development process. Understanding how users perceive products can help designers make decisions that better accommodate user needs. Though several methods have been created that involve the user at different stages of the design process, there is still no clear connection between user perceptions and product improvements. Affordance Based Design (ABD) provides the theoretical background needed to explore such connections. ABD is a systematic design method that uses the concept of affordances to describe the interactions between users and products. The integration of ABD and genetic algorithms (GAs) is proposed as a way to capture the perceptions from users in the form of affordance quality evaluations. This research investigates how those user perceptions can be used to improve or evolve product variants.

A design tool is developed to test product evolution with the proposed ABD/GA integration. The affordance based interactive genetic algorithm (ABIGA) lets designers capture user perceptions of products. In this tool, designers must specify the design parameters of the product as well as some of its affordances. Users can access design experiments from their computers or smart phones and are shown a representation of the product they evaluate. A set of six experiments were carried to test the evolution of a steering wheel. Three of these experiments were done with real users while the rest were done using a random number generator as the input. Two additional experiments were done with real users to test the evolution of a compact digital camera. Results show that product form can be evolved toward better solutions based on the perceptions of users. The re-

sults can also link user perceptions with the form of the product. Designers can extract relationships between affordance evaluations and design parameters. Such relationships can be used to predict how changes in the design parameter values can affect user perceptions of affordance quality.

Product evolution through affordance evaluations could eventually be used to not only improve the external geometry of products, but also certain internal aspects of products. Such a tool could be used in multiple stages of the design process, taking advantage of optimization tools linked to the concept of affordance to automate aspects of the product development process.

Dedication

To my beloved family who gave me all the support, love and hugs. To my friends who give without expecting anything back, without your help I would not have been able to finish this journey.

Acknowledgments

I've been lucky enough to have my whole family with me during this part of my life. In moments when solitude provoked sadness, they were always there to cheer me up by sharing their love. I cannot thank them enough for all they've done. My Parents, Maria and Linguis, have supported me in every decision I've ever made. My siblings, Linguis Jr., Cire and Karen, are some of my best friends, always willing to sacrifice a lot to help me reach my goals.

I would like to thank Dr. Georges Fadel for being my mentor all these years. You're such a kind (and patient) human being; I enjoyed doing research with you and learning from you. Thanks to the rest of my committee, Dr. Gregory Mocko, Dr. Joshua Summers and Dr. Mary Kurz who all gave me insightful feedback on my work. I hope to reach the level of critical thinking that you all have.

I would like to thank all my friends from CEDAR. Your feedback on my work certainly improved its quality. Special thanks to Anthony Garland who shared his coding knowledge and showed me the right path when developing the web application shown in this dissertation. Thanks to my office mates, Qing Mao, Mo Chen, Jingyuan Yan, Meng Xu, Nafiseh Masoudi and Felipe Fernandez for their conversations and sharing their smiles and laughter.

Finally, thanks to the Clemson Judo Club members who not only helped me stay healthy, but also helped me outside of the dojo when I needed help. Special thanks to Sensei Art Gorka, Claire Booth, Tony Rentas and Stephen Smith. I will never forget all your help, without which I would not have been able to finalize my work.

Table of Contents

List of Tables

List of Figures

Chapter 1

1 Introduction

1.1 Background

One of the key abilities that helped our human ancestors advance and evolve was their ability to create and improve artifacts. The design of artifacts can be dated back 1.7 million years ago[1] when early humans created tools out of stone. An important aspect about these artifacts is that like the early humans that created them, they also evolved. Figure 1.1 shows a hand axe made out of stone found in Europe (Meyral, France). This tool features two flat sides and two sharp converging edges with a wide base which served the purpose of a handle. These types of tools were believed to be used for grubbing up and cutting roots and crushing or pounding other objects (Coghlan, 1943). Figure 1.2 shows an evolution of the hand axe. This hand axe variant was made out of bone and was smaller (images not shown in scale); its projectile point shape allowed its creators to

[1] http://humanorigins.si.edu/evidence/behavior/stone-tools/early-stone-age-tools

attach it to spears for hunting purposes. This is an early example of how humans have always sought to improve products by implementing different shapes and materials as newly discovered manufacturing technologies became available.

Figure 1.1 Stone Hand Axe[2]

Figure 1.2 Bone Projectile Point[3]

Nowadays it does not take thousands of years for product iterations to surface. This can be attributed to the need that comes from society to have better things, as well as the creation of design methods, more specifically, systematic design methods. The basic defining feature of systematic design methods is that they decompose the product development process into stages. For example, Pahl and Beitz (Pahl, Beitz, Feldhusen, & Grote, 2006) proposed a systematic design method defining four stages: Planning and task clarification, Conceptual design, Embodiment design and Detail design stages. This method is not only used for *original design*, it can also be used for *variant design*, where

[2] http://humanorigins.si.edu/evidence/behavior/stone-tools/early-stone-age-tools/handaxe-europe
[3] http://humanorigins.si.edu/evidence/behavior/getting-food/projectile-point

the sizes and configurations of product components are varied from a previously designed product. These improvements can be considered evolutions of that product, because the information obtained previously is used to make the subsequent changes.

The motivation to improve a product comes from the awareness that the product can perform better or improve its usability to better respond to user needs. This awareness can only occur if the product is evaluated either by the designer or the end-user. It is important to define what an improvement is based on what is evaluated. The evaluation of a product can include multiple criteria, and if tradeoffs exist, improving all criteria may not be attainable. Nonetheless, improvements can be identified if only a subset of all criteria have been improved. The evaluation feedback is needed to deem the product variant an improvement or not. The Guided Iteration Method (GIM) proposed by Dixon and Poli (Dixon & Poli, 1995) highlights the need of this feedback loop. The basic idea behind this method is that the product needs to be evaluated to check if it meets certain criteria and if the criteria are not met then a *guided redesign* is needed, which is then re-evaluated. This creates an evaluation/redesign loop that ends when satisfactory results are met. The GIM is a tool that can be applied in every stage of the design process proposed by Dixon and Poli: Engineering conceptual design, Configuration design and Parametric design stages. An important distinction between analysis and evaluation is made by Dixon and Poli. Analyses provide quantifiable information based on physical properties. For example, finite element analysis can determine the maximum Von Mises stress on a loaded beam. Evaluations are based on these analyses, and functionality, design for manufacturing, cost, and reliability but can also be based on other criteria which can be subjective. Since the decision of whether the product is acceptable or not comes from the evaluation of the designer based on a variety of criteria, direct user feedback could be used in this process.

An alternate product design method proposed by Maier and Fadel (Maier & Fadel, 2009a) is Affordance Based Design (ABD). Like the methods mentioned previously, this method also decomposes the development process in stages and can also be used to generate product variants (product evolutions). The key difference lies in the concept used to describe the usability of the product. Instead of using the concept of *function*, Maier and Fadel incorporated the concept of *affordance*, which originated from perceptual psychology, in engineering design. The definition of affordances can be simplified to be a representation of interactions, such as the interactions between the product and its user, and the interactions between the components of the product.

As stated earlier, ABD can also be used to generate product variants. The idea is that the interactions between the user and the product (i.e., affordances) can always be improved when a quality characteristic is associated to them. For example, an improvement to a cylindrical knife handle would be a handle that has a triangular cross section because a shape closer to a triangle is formed when the human hand is semi-closed. The interaction between the hand of the user and the handle of the knife is thus improved. The quality of that interaction has been increased with respect to the old handle. The advantage of improving affordances to generate product variants is that if the user is involved in the feedback loop described earlier; this feedback would directly affect the form of the product through the quality of the interactions described.

Up to this point, it has been suggested how the end-user can be involved in the product development process which can be used to improve products, that is, to generate product variants or evolutions. There are multiple methods that can be used to involve the end-user in the development process. Some of the most used methods are: Quality Function Deployment (QFD), User Oriented Product Development, Concept Testing, Beta Testing, Consumer Idealized Design, Lead User Method and Participatory Ergonomics (Kaulio, 1998). These methods are each used in different stages of the product development process. This means that there are multiple ways in which the user could be involved in the product development process.

The goal of user involvement in product development is to better understand user perceptions to design products that will be better accepted once they are released to the market. This of course can be done by choosing a design method and coupling it with an existing method for user involvement or even creating an entirely new user involvement method.

1.2 Research Problem

Though there are many product development methods, some methods are better platforms for user involvement than others. This is the case with Dixon and Poli's design method which has a tool (Guided Iteration Method) that can be used to create product variants and can be modified to involve the user in the evaluation of products. Affordance Based Design has the advantage of using the concept of *affordance* which is used to describe the usability of the product and at the same time relates the user with the product;

this suggests that users should intuitively be able to evaluate the affordances of a product. But which design method is more suitable for the creation of product evolutions guided by user evaluations? As suggested earlier, the answer to this question will depend on being able to identify which user involvement method can be coupled to the product development method.

Assuming a product development method and a user evaluation method have been chosen, what do users evaluate and how is that feedback used to generate product variants? So far it has been suggested how users might be able to evaluate the quality of their interactions with the product through the concept of *affordance*, but the way that information is used to generate new product variants is still unknown. Another important aspect of having the user in the development process is the challenge of reaching the user. In the methods presented earlier the physical presence of the user was necessary. This of course limits the amount of users that can be accessed.

The aim of this dissertation is to give designers a platform that streamlines the creation of product variants with user evaluations. To this end, user involvement methods might need to be modified or created to accommodate specific characteristics of a product development method. Such a platform would help designers find product improvements validated by the users of the product themselves. Moreover, a platform of this kind would accelerate the developments process if the user-access problem exposed earlier is solved.

As a consequence of creating the platform described earlier, this dissertation also investigates how user feedback is used to create product variants. More specifically, how the information gathered from users can be directly used to affect the form of products and potentially use that information in future design projects.

1.3 Dissertation outline

The following provides an overview of this dissertation's chapters and their content.

Chapter 2 starts with background literature on how different design methods support product improvements. The second section provides a literature review of the different methods used to involve users in the product development methods and how user feedback could be used to improve products. The third section explores how user feedback is used to evolve product variants and classifies the different methods in two types. The first type of methods indirectly uses the user feedback through a mathematical model of their preferences. The second type uses that feedback directly to affect product form by the means of optimization tools. The last part of the chapter summarizes the gaps in the literature and presents the focus of this research through hypotheses and research questions.

Chapter 3 presents how the concept of affordances is integrated with interactive genetic algorithms to support the evaluation of affordance quality by users. Sections two, three and four provide details about the development of the design tool (a web application) that integrates the two concepts. Details about the different elements of the application and their use are given in the following sections. Finally, general steps on how to setup evolution experiments in the application are given.

Chapter 4 gives the results of a series of evolution experiments performed with real users. The first section provides the method followed to test the hypotheses given in chapter one. Section two provides de results of each experiment and shows how the data was analyzed to answer the research questions. The results are then discussed in the same section. Finally, details on how the results could be used in the design process are given.

Chapter 5 explores the results of a separate set of design evolution experiments. These experiment results are used to analyze the effect that the choice of affordances and design parameters has on the evolution of products. Guidelines on how to select affordances and design parameters for evolution experiments are provided based on the insight gained from the analysis of the results. The last section of this chapter provides guidelines on how to check the results of product evolutions for dependency between the different affordances of the product.

Chapter 6 summarizes how the results address the different research questions provided in the early chapters of this dissertation. A list of contributions to the engineering design community is given to emphasize how the different tools generated in this research could have a broader impact. Finally, future work is explored which details possible paths this research could take to expand the understanding of product evolution through affordance evaluations.

Chapter 2

2 Literature Review

This chapter provides an overview of how users can be involved in the product development process using different methods and techniques. The first part of the literature review introduces some of the well-known systematic engineering design processes and how these platforms can support the improvement of products. User involvement methods are then are presented in the second part of the literature. These methods can be seen as design tools that can be implemented in a systematic design method. The third part of the literature introduces techniques that can be used to process the feedback obtained from users in a more direct manner. The last section of the chapter goes over the hypotheses and research questions that define the focus of this research.

2.1 Systematic Engineering Design Methods

One of the defining characteristics of systematic engineering design methods is that they decompose the product development process in design stages. The design process is commonly separated in four different stages: problem understanding, conceptual design, embodiment design and detail design. Having different stages throughout the design process allows designers to constantly check for problems or inconsistencies and be able to iterate. This lets designers fix problems before they become too difficult to solve. Even if errors are found during the later stages of the design process, systematic methods help designers identify in which stages those errors originated.

Compared to an ad hoc approach to design, the iterating nature of systematic design processes facilitates the improvement of products. As previously suggested, improvements can be done during the development process by iterating within design stages or iterating across design stages. These improvements can also be done on finished products, by creating design variants. Product variants are improvements of already designed products where sizes and arrangement of parts are modified to improve their performance or usability (Pahl et al., 2006). Though there are many methods that adapt to the descriptions provided earlier, a well-known systematic approach is the method proposed by Pahl and Beitz.

2.1.1 The Pahl & Beitz Design Method

The systematic approach to engineering design proposed by Gerhard Pahl and Wolfgang Beitz was created, among other reasons, to provide re-usable design solutions as well as serve as a platform that encouraged computer support (Pahl et al., 2006). The product development process is divided into four main phases:

- Planning and task clarification
- Conceptual design

- Embodiment design
- Detail design

The purpose of the first stage (*planning and task clarification*), is to gather information about the performance and usability expectations of the product. The outcome of this stage is a requirements list which is continuously updated to reflect the current understanding of the design problem. Two types of requirements are defined: demands (constraints) and wishes (criteria). In the *conceptual design* stage the principle solution is found. It is at this stage that a key concept is used, and it is that of *function*. Functions are transformation of energy, material and signal inputs. Function structures are created which describe the tasks that the product and its sub-systems perform to fulfill the requirements set in the initial stage. Working principles are then identified which define how functions are fulfilled. These principles are combined into solution variants which later go through a selection process. The solution variants that make it through the selection process then go through an evaluation process. The concept that better meets a set of criteria is finally chosen as the principle solution. The principle solution is the input to the next stage, *embodiment design,* where a layout is specified. It is at this stage where the form of the product is specified. The last stage is *detail design* where production instructions are created. This involves detail drawings, production and assembly instructions.

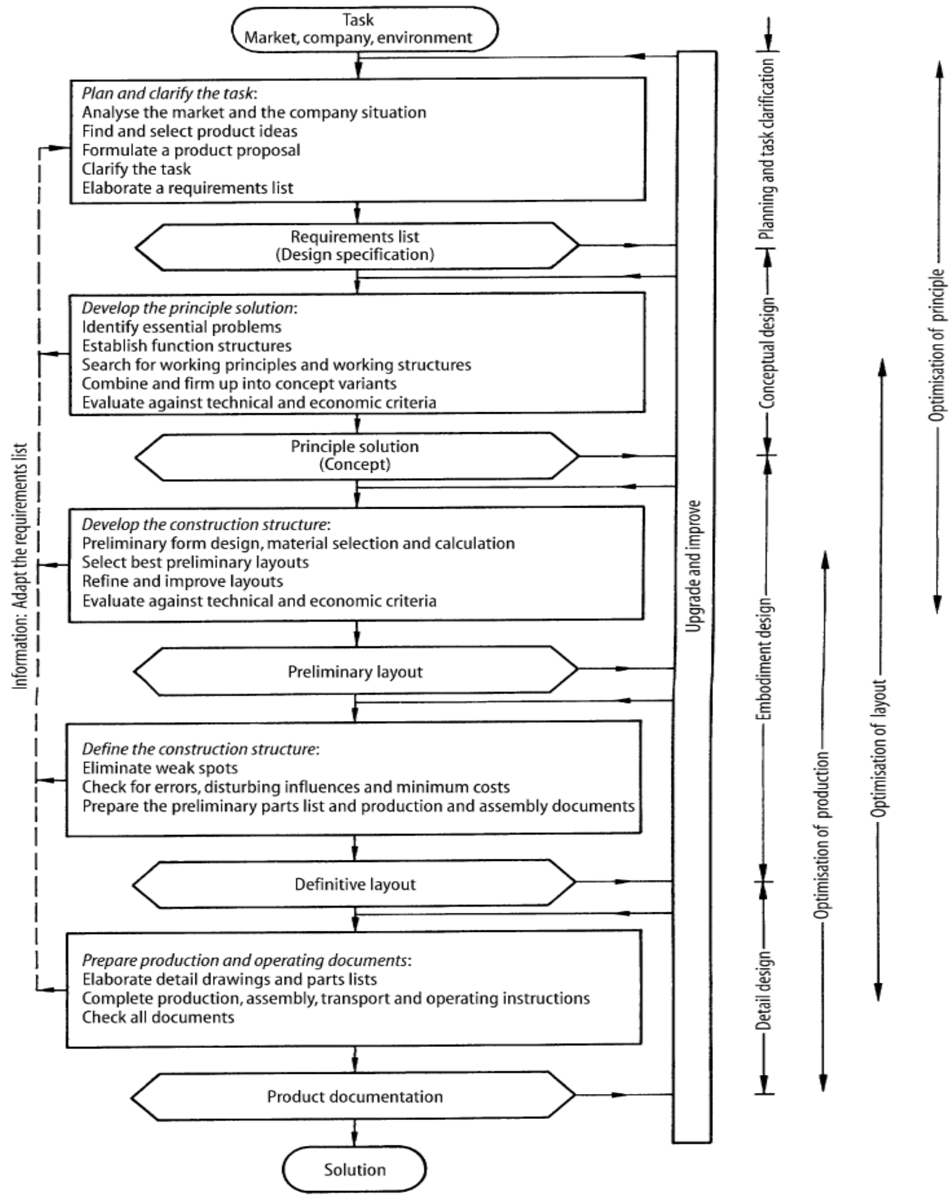


Figure 2.1 The Pahl & Beitz Systematic Design Method (Pahl et al., 2006)

Figure 2.1 shows the different stages of the product development process as described earlier. Of particular importance is the emphasis that is given to upgrading and improving the outcomes of each design stage. In the task and clarification stage, improvements are reached when a better understanding of the problem is obtained. This could be in the form of more detailed descriptions of the current requirements or by finding new requirements altogether. During the conceptual stage the improvements can be attained by finding new working principles (new discovered technologies) which could potentially help create a principle solution that better meets the design criteria The improvements done in the early stages of the design process affect the overall shape (or topology) of the product, because additional requirements might mean that more subfunctions need to be added to the function structure. Changes to the working principles could also directly affect the shape of the product. It is only at the embodiment stage where improvements to the form of the product can be made without changing its solution principle.

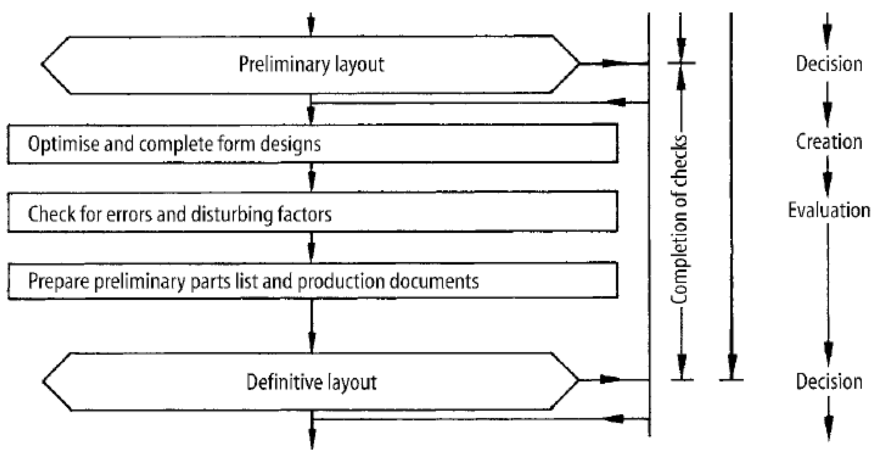


Figure 2.2 Pahl & Beitz Embodiment Design Steps (Pahl et al., 2006)

As seen on Figure 2.2, the form of the product can be optimized. The optimization process of course cannot be performed without a form of feedback, which is why evaluation must be performed every time a change is made. The type of evaluation proposed by Pahl & Beitz involves technical and economic criteria. This includes space requirements, material usage, strength, human-machine relationship (ergonomics), ease of assembly, and costs, among others.

The inclusion of ergonomic criteria in the evaluation of design embodiments highlights the importance of designing products with the interactions between the user and product in mind. This is made obvious in the Pahl & Beitz framework as one of the embodiment design guidelines is that of *design for ergonomics* (DFE). The DFE guidelines consider biomechanical (body posture and movements, body dimensions), physiological (loads, stress and fatigue) and psychological (how information affects user behavior) issues. However, the guidelines provided do not involve the user directly in the product development process. An example of direct user involvement would be to have the user do product evaluations and be a part of the improvement of the product.

2.1.2 The Dixon & Poli Design Method

The Pahl and Beitz design method studied earlier showed the benefits of segmenting the product development process and how this can support product improvement. John Dixon and Corrado Poli also created a systematic approach to product development (Dixon & Poli, 1995). Their method also dissects the product development process in four stages:

- Engineering conceptual design
- Configuration design
- Parametric design
- Detail design

In principle, these stages are similar to the Pahl and Beitz design stages. The *engineering conceptual design* stages combines the planning and task clarification stages with the conceptual stage found in the Pahl and Beitz framework. The specification of requirements is included in this stage, as well as the creation of function structures from which basic forms are created to fulfill those functions and requirements. In the *configuration design* stage the approximate form of the different parts and their configuration is determined. It is only at *parametric design* that the final form of the product is found. This is accomplished by the application of different engineering analysis tools such as calculations based on physical principles and optimization techniques.

The Dixon and Poli method also emphasizes how improvements can always be made to the outputs of each design stage. However, Dixon and Poli created a separate design tool that can be used in at any design stage to achieve improvements. The Guided Iteration Method (GIM) (Dixon & Poli, 1995) is comprised of 4 stages (see Figure 2.3): Problem formulation, Generation of alternatives, Evaluation of the alternatives and Guided redesign. The *evaluation of the alternatives* is done by performing engineering analysis and using the results to check how the engineering requirements are met. The outcome of the *redesign stage* can be seen as an evolution of the product and is done by considering the evaluation results of the previous product alternatives.

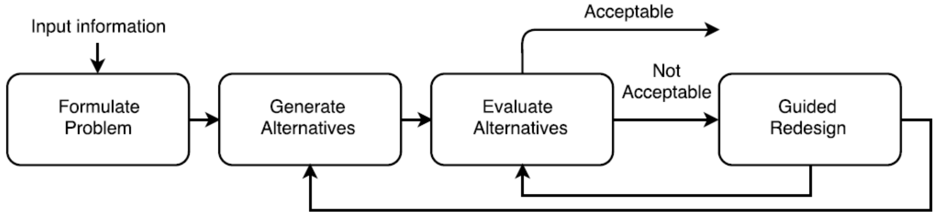


Figure 2.3 Dixon and Poli's Guided Iteration Method

As in the Pahl and Beitz framework, the Dixon and Poli method does not provide solutions where the user of the product is directly involved during the development of products. This is not to say that these frameworks do not consider the interactions between the user and the product. As a matter of fact, both methods make use of the Quality Function Deployment (QFD) to map the needs of customers into engineering specifications. However, the user is only involved during the initial stages of the design process and the feedback obtained from the users is not directly used to generate product variants. Nonetheless, the GIM opens up the possibility of including the user in the feedback loop. The user could become part of the evaluation step, evaluating the different product variants that are created.

2.1.3 The Maier & Fadel Design Method

The design methods shown so far are good examples of how a systematic approach helps designers improve their products (or design concepts). The idea of including the user in the improvement loop was also suggested and it was hypothesized that the user could be used to evaluate the products themselves. There still remains a question about what the users would be evaluating. The criteria used in the design methods shown earlier are objective engineering analyses for which user input is not needed. Though there are many product development methods, some methods are better platforms for user involvement than others. Affordance Based Design (ABD), introduced by Johnathan Maier and Georges Fadel (Maier & Fadel, 2009a), has the advantage of using the concept of *affordance* which is used to describe the possible ways to interact with or use the product. This suggests that users can evaluate the affordances of a product by observing it.

The term *affordance* originated in the field of perceptual psychology. It was introduced by Gibson (Gibson, 1979) who defined it as "The affordances of the environment are what it offers the animal, what it provides or furnishes, either for good or ill." It was created to describe what a system (e.g., an artifact) provides to another system (e.g., a user). Norman then extended the term to aid in the design of consumer products (Norman, 1988), but did not incorporate the concept of affordance as fundamental to the design of *any* artifact. Maier and Fadel (Maier & Fadel, 2009a) introduced the concept of affordances as being fundamental to engineering design and defined it as a relationship between two subsystems in which potential behaviors can occur that would not be possible with either subsystem in isolation.

An advantage ABD presents is the way it links the design entities that interact in the design process. The idea of affordances allows us to describe the relationships between these entities. Affordances determine how the artifacts can be used; it is the designers' task to design artifacts by identifying relevant affordances and anticipating how the artifact being designed should be used by the user (Maier & Fadel, 2009a).

To better understand the concept of affordances, its properties need to be known. According to Maier and Fadel (Maier & Fadel, 2009a), the following are the properties of affordances: *Complementarity*; it tells us how the interaction of two design entities is described by an affordance and that the affordance cannot exist without the interaction of such entities. Sometimes some features of an artifact may cause harm to the user, this means that some affordances can also have a negative impact, thus, affordances have *polarity*. Knowing this, the designers should concentrate in identifying not only positive affordances but also the negative ones, in order to reduce the latter. We also have to be aware that an artifact will have multiple affordances (*multiplicity* property), and these affordances can either be positive or negative. *Quality* tells us how well a system affords a specific use or action based on the perception of the user. The user perception can be positive or negative with varying levels of intensity. With this property in mind, designers know that there could always be a better design solution to a design problem; for example, *sit-ability* has a better quality in chairs than in stools. The last of the affordances' properties is that they are *form dependent*: it is the structure of artifacts that determine what they afford a user and/or other artifacts. This affordance and artifact dependency also suggests that relationships between affordances and design variables exist.

Depending on which entities are interacting, three types of affordances have been identified, two of these were proposed by Maier and Fadel (Maier & Fadel, 2009b), *artifact-user* affordances (AUAs) and *artifact-artifact* affordances (AAAs). Hu and Fadel (Hu & Fadel, 2012) later proposed an additional type of affordance, *artifact-environment* affordances (AEAs). Other authors have also categorized affordances differently (Galvao & Sato, 2005; Gaver, 1991; Hartson, 2003; Kannengiesser & Gero, 2010; McGrenere & Ho, 2000; Pols, 2012; Raubal & Moratz, 2008), but some of these categorizations are applied on other fields such as artificial intelligence. Artifact-user affordances are interactions between the user and the artifact in which properties of the artifact may be perceived to be useful or detrimental to the user. In a similar way, artifact-artifact affordances are interactions between artifacts where such interactions are possible due to specific properties of each artifact. There are some entities that are not considered to be users nor artifacts (e.g., substance, medium and natural objects) but still interact with the artifact; hence, the environment has also been considered an entity in ABD (Hu & Fadel, 2012). Artifact-environment affordances are interactions between the artifact and external environment entities that affect the performance of the artifact.

Each affordance type can be further classified in *doing* and *happening* affordances (Hu & Fadel, 2012). This helps to define the direction of the action that the affordance represents. For example, the *turn-ability* of a door knob would be a *doing* type of affordance because the user performs the action of turning; whereas the heating-ability of a car seat is a *happening* type of affordance because the heating "happens" to the user.

The designer-artifact-user (DAU) entities form a complex system (Maier & Fadel, 2009a). The entities, or subsystems, are related in such a way that the designer is capable of modifying the properties of the artifact by specifying a set of AAAs, AEAs, and AUAs.

Maier and Fadel's approach to design is also systematic (Maier & Fadel, 2003), Figure 2.4 shows the steps that need to be followed to design an artifact from scratch using the concept of affordances. This method is similar to the systematic methods shown earlier in the sense that concepts are determined first and then are followed by embodiments. The main difference with respect to those methods is the use of the concept of affordance. Only AUAs are initially defined, that is, affordances that describe the interaction between the user and the product. This information can of course be determined from a set of requirements provided by a client. Design concepts are then created based on the affordance structure generated in the previous step. These concepts then go through a selection process which considers how the different concepts satisfy the different affordances. The concept that offers the highest affordances qualities with the least amount of negative affordances is chosen for the embodiment stage. The selected concept could not be embodied if the interactions between its components are not defined. This is why the AAAs and AEAs are determined. The final step of the ABD process is to design each affordance, for which a method is given by the authors.

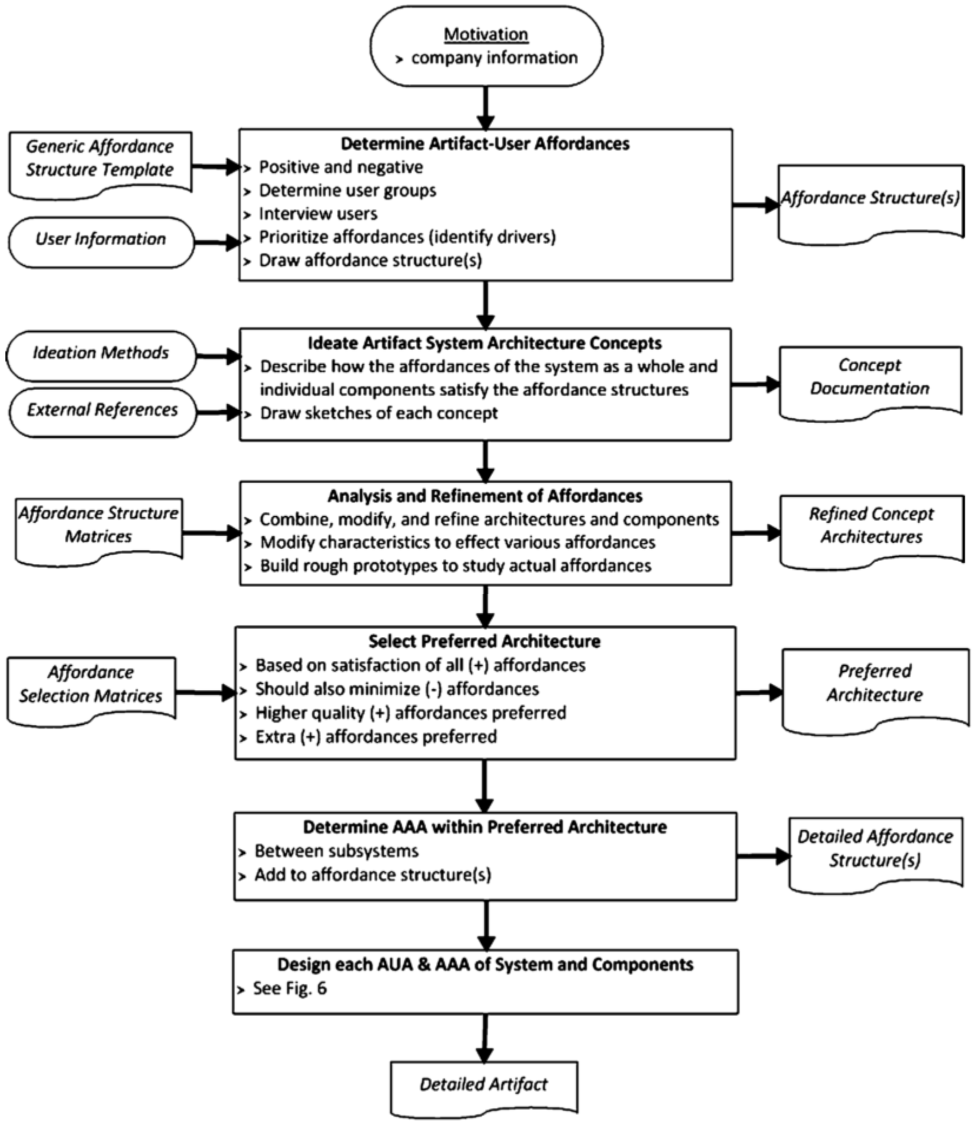


Figure 2.4 The ABD Method (Maier & Fadel, 2009b)

It was suggested at the beginning of this section how users could evaluate the affordances of products. This can be accomplished through the evaluation of one affordance property. As described earlier, *quality* tells us how well a system affords a specific use or action based on the perception of the designer or user. This perception can be positive or negative with varying levels of intensity. For example, both a chair and a briefcase offer sitting-ability, but because of the back support (as well as thigh support) that a chair offers, it can be said that the *sitting-ability* of the chair is better than the *sitting-ability* offered by a briefcase. If the user becomes the evaluator of the affordances of the product and a feedback loop is created, design variants could be improved.

Maier and Fadel have suggested the use of Affordance Based Design to redesign products (Maier & Fadel, 2009b), or in other words, to improve products. They provided methods and tools to improve the affordances of products. Analyzing the quality of an artifact-user affordance (AUA) may point to the need to improve it. The quality of the affordances can be assessed by either the designer of the artifact or by different types of users (e.g., manufacturing, maintenance, end users) (Maier & Fadel, 2009a). In their examples the authors did not specifically use affordance quality analysis to determine which affordances needed improvement. Instead, they used tools such as the Affordance Structure Matrix (ASM). The ASM is a matrix that relates the affordances of the product with its components (Maier & Fadel, 2007). This tool lets designers direct their attention to the affordances that have the least number of positive relationships or the most number of negative relationships. Once these affordances have been identified the designer can redesign specific affordances with methods found in the literature (Maier & Fadel, 2009b).

Maier and Fadel are not the only ones that suggest that products can be improved with ABD. Kim et al. created an affordance repository that could be used to redesign products (Kim, Shin, Kim, Noh, & Kim, 2012). The repository contains features that provide the targeted affordance and designers could use those solutions and implement them in their designs. In their experiments they had a group of designers redesign a coffee tumbler. The aspect of quality analysis can be seen in the way that designers selected one affordance feature from many that were available. The designers were capable of judging which affordance features were better for the redesign of the tumbler.

It wasn't until Gaffney et al.'s research that affordance quality analysis was directly used to describe the improvement of products over time. Gaffney et al. (Gaffney, Maier, & Fadel, 2007) found that the evolution of products can be explained with an improvement in the quality of its individual affordances; even if the products were not designed using the Affordance Based Design (ABD) method. Gaffney et al. described the evolution of vacuum cleaners by performing affordance quality analysis on evolutions of the product. The affordance quality analysis was performed by the researchers themselves. Even though they do not mention it, it is evident that: t*he affordance quality assessment by designers or users is naturally subjective*. A person could agree that the *sitting-ability* of a chair is better than the one provided by the briefcase, but could still use the briefcase's *sitting-ability* not minding the inferior quality it offers. On the other hand, there could be another person that won't tolerate the *sitting-ability* offered by a briefcase and would rate the chair's affordance significantly higher.

Evolving products through the improvement of their affordances as perceived by end-users would be a way to directly use their input in the design/redesign of products. Using the input from end users would help the designer know what users perceive to be a high quality product. The challenge then is how to utilize that feedback directly in the design process.

2.2 User Involvement Methods

The Pahl and Beitz method encourages iteration to improve the outcomes of each design stage by evaluating technical and economic criteria. The Dixon and Poli method makes use of the guided iteration method and can be used at any stage of the product development process. The Maier and Fadel method can be used to redesign products by improving the quality of their affordances. Though these design methods have a platform that encourages product improvement they were not created to involve the user directly in the design process.

There are of course many other product development methods; some were created specifically to improve products. Otto and Wood's redesign framework (Otto & Wood, 1998) emphasizes the importance of customer feedback in the evolution of products. In fact, the understanding of customer needs is part of their task clarification stage of the redesign process. However, as is the case with the other design methods studied, the user feedback obtained in their framework is not used directly in the design process. This information is interpreted by the designers using tools such as the House of Quality (Maier & Fadel, 2007). Design tools that allowed direct user involvement had to be created to improve the designer's ability to interpret user feedback.

There are many user involvement methods and Kaulio classifies them according to the level of involvement the users have in the product development process (Kaulio, 1998): '*design for'*, '*design with'* and '*design by'* methods. A representative 'design for' method is that of quality function deployment (QFD). QFD makes use of the house of quality (HOQ) to translate customer needs into technical requirements (Pahl et al., 2006).

The user is only involved during the initial stages of the product development process; the feedback has to be interpreted by the designer early in the design process. This means that the feedback obtained from the users cannot possibly be used to directly affect the shape of the products because the users are not evaluating solution embodiments.

The 'design with' user involvement methods have users do evaluations on product concepts or prototypes, using their preferences as the main input during the product development process. The *User Oriented Product Development* is one of these methods that originated in the field of ergonomics engineering (Rosenblad-Wallin, 1985). Similar to QFD, this method analyses user needs and interprets them into engineering requirements. The interaction with users is done through formal interviews, questionnaires, user panels or observation methods (among others). This method however goes a step further by having users evaluate prototypes of the solution. Design variants are then created until a solution that better meets user requirements is found. Some methods have users assess concept solutions rather than embodiments; *concept testing* is one of these methods where users evaluate concept sketches, mock ups or early prototypes (Moore, 1982). User involvement methods also engage the user during the later stages of the product development process; such is the case with *beta testing*, commonly used in software design (Dolan & Matthews, 1993) . In beta testing a working prototype is given to users. Designers then test the level of satisfaction (through direct conversations with users or through questionnaires) and make modifications accordingly. The last user involvement method type is the 'design by' methods. Users take the role of the designer in the development process, generating engineering requirements (Ciccantelli & Magidson, 1993) and design concepts (Herstatt & Hippel, 1992). Although the 'design by' methods may seem to take a long time to provide results due to the increased involvement of users in the design activities, recent computer design tools might improve the way the information is handled to generate results faster (Ramanujan, Vinayak, Nawal, Reid, & Ramani, 2015).

Based on Kaulio's categorization, having users evaluate the quality of affordances would fall under the 'design with' type of user involvement. This is because these methods have an iterative approach, that is, feedback is extracted from the users and then changes are made until the feedback is deemed positive based on how the product meets the evaluating criteria. The 'design for' methods do not support this iterative scheme, because they are generally applied during the design specification stage and are too far apart from the embodiment stage of the design process.

Though these methods seem to effectively involve the user in the design process there is a major drawback to their approach: It can become a lengthy process. Not only because the physical presence of the user is needed for these tools, but also because there is not automated way to capture and process the feedback. The lack of computational support in these methods made researchers look into other ways to speed up the development process when the user is involved.

2.3 Product Evolutions and User Involvement

Based on the literature shown thus far, the main goal of including the user in the development process is to design or improve products with feedback that reflects their preferences. Product improvements, or product evolutions, can be found with a variety of methods that process the feedback from users (like the ones shown in previous sections). The problem with these types of input processing techniques is how long it takes to effectively create product evolutions. This is why researchers started creating techniques that allowed designers to come up with design evolutions at a much faster rate while focusing on the elicitation of user preferences.

Product evolutions can be can be achieved with *direct* or *indirect* user involvement; it depends on whether the feedback obtained from users is used directly to create new product forms or if that feedback is used to create models that estimate users' assessments (preference models). Both approaches are explored next and some examples are given.

2.3.1 Indirect User Involvement Methods

The key defining feature of indirect user involvement methods is that user feedback is interpreted in a mathematical model. The fact that the preference model of users can be represented in a mathematical function is what speeds up the evaluation process.

User preference models can be used in optimization. An example of how this can be achieved is given by Reid et al. The feedback obtained from users is in the form of perception based attributes assessments. Perception based attributes are the design properties that can influence people's judgements about objective qualities of a product (Reid, Frischknecht, & Papalambros, 2012). The way Reid et al. used perception based attributes in optimization is through a perceived environmental friendliness (PEF) model for vehicle silhouettes (Reid, Gonzalez, & Papalambros, 2010). The model, which represents users' preferences, was obtained by showing vehicle silhouettes to multiple users and asking them questions about it. The feedback from the users was analyzed against the parameters that define the shape of the vehicle silhouette. The most significant parameters were identified and the mathematical model was obtained through regression analysis. The optimization problem had fuel economy (FE) as the main objective (a model that depends on the vehicle's design parameters) and used the PEF model as one of the constraints. It is important to distinguish the process of obtaining a preference model and the optimization process as being separate. In Reid et al.'s approach, the optimization experiments themselves do not need real users as an input because of the use of the PEF model.

Another example is given by Orsborn and Cagan. They (Orsborn & Cagan, 2009) used a multi-agent shape grammar implementation to generate product forms with a user-preference function. Shape grammars are used to generate geometry variants from a base geometry sample. Different shapes can be created which define all the basic forms of the base geometry. Shape rules can be created by defining how the basic forms are related to each other. These rules are comprised of a left and a right side. When the shape on the left matches a shape in a drawing, the rule is applied, and the shape is changed to match the shape on the right side of the rule. Addition and subtraction of shapes allow shape modifications, in other words, shape variants that go beyond individual shape rules. The method to generate variants is comprised of three main parts: A shape grammar interpreter that modifies the shape of the product by implementing the shape rules, an agent system that decides which shape rules to apply while generating and optimizing product designs and a preference investigator that maps the product designs to user preferences using a preference function. The preference function is found through surveys administered to potential customers.

The preference function, or utility function, plays an important role in linking the preferences of users with the shape of the products. This is a key aspect to quickly create product variants based on the input from users because user evaluations would directly affect the form of products. Orsborn et al. (Orsborn, Cagan, & Boatwright, 2009) employ utility functions to map product shape to user preferences. The shape of the product is defined by Bezier curves that describe individual characteristics of a basic product shape. The input from users is obtained through discrete choice conjoint analysis surveys where users select pictorial representations of products as opposed to verbal descriptions of the products. A major drawback of using this approach to define user preference models is that they approximate the preference of individual users only. This means that if these preference models are used to generate designs, the results would only reflect the preference of individual users.

2.3.2 Direct User Involvement Methods

Though indirect user involvement methods represent a faster alternative to create product variants than the methods presented in earlier sections, finding the preference model can be time consuming. Researchers looked into different ways to create product variants with the input from users without the need of preference models. Direct user involvement methods make use of algorithms that instantly process the input from users to create product variants.

2.4 Genetic Algorithms and product evolution

Genetic Algorithms (GAs) have been used extensively in engineering design to evolve products (Bentley, 1999; Gen & Cheng, 1997). GAs are search based algorithms that mimic the mechanics of natural selection and natural genetics (Goldberg, 1989). There are three basic evolutionary operators in GAs: *Reproduction*, *Crossover* and *Mutation*.

GAs start with a randomly generated set of solutions called *population*, where each solution parameters are encoded in a *chromosome* (Gen & Cheng, 1997). The chromosomes, or solutions, *evolve* through sequential iterations (*generations*). In each generation the solutions are evaluated by a *fitness* function and a subset of the population is selected to create new solutions for the next generation. The solutions with better fitness scores have higher chances of being selected. This step is part of the *reproduction* operator in GAs. The *crossover* and *mutation* operators are functions that generate new solutions using the solutions that were selected by the *reproduction* operator as the input.

Solutions are generally encoded in GAs, meaning the operators don't necessarily work with real valued variables. **Figure 2.5** shows how a basic *crossover* operator works to create two new solutions (*offspring*) denoted as C1 and C2 from two *parent* solutions denoted as P1 and P2. The solutions are encoded as binary bit strings in the figure. A random position is chosen where the parent solutions are cut and recombined to generate the offspring.

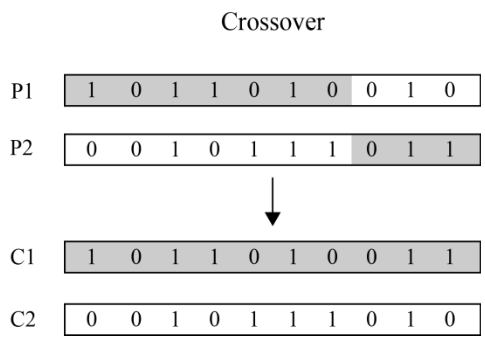


Figure 2.5 A basic Crossover Operator

Mutation is another GA operator that creates new solutions. When the solutions are encoded as binary bit strings, a simple mutation operator randomly selects a position in the binary bit string and switches its binary value, effectively creating a new solution. The sequence in which the GA operators are used is shown in the pseudo code shown next, adapted from (Gen & Cheng, 1997).

GAs basic structure (Gen & Cheng, 1997):

```
Begin
 t ← 0;
 initialize P(t);
 evaluate P(t);
 while (not termination condition) do
    recombine P(t) to yield C(t);
    evaluate C(t);
    select P(t+1) from P(t) and C(t);
```

```
        t ← t+1;
    end
end
```

As described earlier GAs require a function that evaluates the solutions to determine their fitness. If the evaluation is subjective, the human evaluating function becomes difficult to model. This is because the evaluation is based on personal preference or values. However, given a proper interface, the evaluation of solutions can be manual, that is, humans can evaluate GA solutions. This is how researchers incorporated GAs to create direct user involvement tools.

2.4.1 Interactive Genetic Algorithms

Interactive Genetic Algorithms (IGAs) are genetic algorithms where the evaluating function is substituted by human users that interact with the GA through an interface. IGAs have been used in a wide variety of applications, for example, music, graphic art, industrial design, speech and image processing, virtual reality, controls and robotics to name a few (Bentley, 1999; Takagi, 2001).

Banerjee et al. (Banerjee, Quiroz, & Louis, 2008) implemented IGAs to evolve floorplans and widget layouts. They achieved satisfactory results within 15-20 generations/iterations. Banerjee et al. found that designs evolved by a collaborative peer group were consistently rated higher on the "originality" scale when compared to designs evolved by a single designer. They also used a collaborative framework where individual

IGA sessions can connect to each other to send and retrieve concepts. This means such setups use multiple IGAs; each user sharing one "best" solution per generation. The users can select solutions from other users and incorporate them in their own population.

Brintrup et al. (Brintrup, Ramsden, & Tiwari, 2007; Brintrup, Takagi, Tiwari, & Ramsden, 2006) used IGAs to optimize manufacturing plant layout designs. There were two types of IGAs: a sequential and a multi-objective IGA. The sequential IGA could be set to optimize using quantitative and qualitative objectives sequentially. The multi-objective IGA considered both qualitative and quantitative objectives simultaneously. In either case the user was responsible for the qualitative assessments of all individuals of each generation. Their results showed faster convergence when optimizing quantitative and qualitative objectives at the same time.

Brintrup et al. (Brintrup, Ramsden, Takagi, & Tiwari, 2008) also did experiments where they used an IGA to have users assess two subjective design parameters (comfort and liking). One of the challenges of using IGAs is the large number of evaluations that need to be done by users (Hsu & Chen, 1999; Takagi, 1998). However, as (Banerjee et al., 2008) point out, it is beneficial to have multiple users do these evaluations since they diversify the results, effectively reducing the number of evaluations that a single user would need to perform. Researchers have worked on solutions to this problem (Hsu & Chen, 1999), but these solutions involve approximating users' input for the concepts they don't assess.

Ren and Papalambros created an alternative to the way IGAs search for optimal solutions (Ren & Papalambros, 2011). In their framework, users provided a binary type of feedback, they either preferred a solution or not over other solutions shown. The algorithm used is called 'efficient global optimization' (EGO) search and was used in a vehicle styling design problem. They tested their algorithm by having users select vehicle shapes according to a target shape. Their results are interesting regarding what users do in terms of assessing product shapes. Though their results did not converge based on the Euclidean distance between the optimal solution and the target shape, visual inspections of the results showed that users did in fact select shapes that were close to the target shape (users were focusing on the shape of the roof of the vehicles). The authors speculate that this might be due to users not comparing shapes in the design variable space but rather in a feature space of the product. Their results suggest that users focus on different aspects of products when evaluating them and not the product shape as a whole.

2.5 Research Questions

The systematic design methods like Pahl and Beitz's paved the way to create product evolutions by the means of iteration. Other methods were created to include the user in the development process by having them generate design requirements, assess solution concepts and even prototypes. These methods were proven to be effective; but the way the feedback is processed is not efficient because it resembled the activities of the designers themselves, which could translate to lengthy design processes. To improve the time it takes to process the feedback obtained from users, researchers started using optimization tools such as search and evolutionary algorithms. From this two ways to process user feedback can be recognized: methods that capture user preference in a mathematical model and the methods that do not. As shown in the literature, the methods that do not need a preference model have the advantage of being able to directly process the feedback obtained from users.

Most of the methods that involve the user in the design process lack a way to relate user feedback with the shape of the product. There has been a lack of computational tools that help designers find relationships between the geometry of the product and user perceived qualities (Orbay, Fu, & Kara, 2015). This would give designers the power to predict how design changes would affect user perceptions of products. A possible reason why this has not been explored is because the criteria that users evaluate cannot be naturally related to product shape. ABD's concept of affordance might be able to bridge this gap. The focus of this research is given with more detail in the form of research questions and hypotheses shown next.

RQ1. Can design variants be evolved using an affordance-genetic algorithm integration that uses end-users' input?

Product variants can be evolved using an affordance-genetic algorithm integration that uses real end-users to evaluate the quality of product affordances. Having multiple end-users assess the quality of the affordances of product concepts can help us evolve them toward better perceived solutions. A graphical user interface can be created where real end-users can see the design concept in question along with the designer's targeted affordances and be able to evaluate them.

RQ2. Can relationships between affordances and design parameters be extracted from design evolution experiments results?

RQ2.1. Can affordance and design parameters relationships be used to predict user assessments?

Relationships between the affordances and the design parameters of the product can be extracted from the evaluations of end-users. The evaluations of the quality of the affordances of a product can be related to the changes in the values of the design parameters of the design concepts. This information can be used by the designer to target specific aspects of the product by selecting affordances that show to have an effect on changing the design parameter values toward values that yield better quality design variants.

It is expected that not all types of affordances can effectively be evaluated in a virtual environment. Some affordances require the use of different human senses to be evaluated and not all senses can be involved in a virtual environment. However, *usability aspects of products can be assessed by judging the size of products*. For example, people can assess that the *grip-ability* of a half-liter coke bottle is better than the *grip-ability* of a two-liter bottle just by looking at the objects. This is also true if only pictures of the bottles are shown (provided a size reference is also given).

Guidelines can be created to increase the chances of a successful product evolution experiment using the affordance-genetic algorithm integration tool. The selection of affordances to be shown to users might have an effect on the outcome of the evolution experiment. Also, the way the product is presented to users is expected to influence the users' ability to judge its affordances, because users need to be able to compare the size of the product with the size of their bodies and other objects that may be present in the environment. This can be achieved indirectly by showing the product next to objects that users are familiar with.

The next chapters investigate how the integration of the concept of affordance with optimization tools can be used to capture user perceptions in the form of affordance quality assessments.

Chapter 3

3 The Affordance Based Design and Interactive Genetic Algorithm Integration

IGAs can provide the platform needed for the evaluation of affordance quality to evolve products. Nguyen et al. (Nguyen, Guarneri, Fadel, & Mata, 2012) found a way to integrate the concept of affordances with genetic algorithms (GAs) and make physical characteristics of objects change according to the evaluation of the quality of their affordances. Nguyen et al. used a modified version of the Affordance Structure Matrix (ASM) to evaluate the quality of the affordances of a product. As described in Chapter 2, the ASM was created as an attention directing tool to let designers know which affordances or product components need improvement. Instead of describing the product by its components, Nguyen et al. described the product with its design parameters. Though the most important change was made to how designers quantified the relationships between the design parameters and the affordances. The original ASM only let designers specify if an affordance is related to a component or not. The modified matrix allows entering discrete values (from -3 to +3). This evaluation allows designers to judge how well the product provides specific affordances, in other words, it lets designers rate the quality of the affordances of the product.

The basic idea of the ABD/GA integration is to look at the evolution of products as an optimization problem where the quality of the affordances is being maximized. As previously mentioned, the need for including user input into the affordance based design process is to know what the quality of the affordances are from the perspective of the end-user. The solutions are defined by the set of design variables values and are encoded in such a way that the GA is capable of doing its evolutionary operations on them. Table 3.1 shows an example of how two steering wheel concepts, defined by eight design parameters, are encoded as binary strings. For example, the first parameter could adopt three values {2, 3, 4}, for which three binary representations are needed {00, 01, 10}. The solution representation can also be real-valued, meaning real numbers are used by the GA operators. This is explained in the next section, as it is how the affordance/genetic algorithm integrations represents solutions in this research.

Table 3.1 Binary Encoding of GA Solutions (from Nguyen et al. 2011)

Design Parameters String	**Design Parameters Binary String**
3|180|1|1|3|1|1|0	010111100001000000
3|160|3|2|2|1|2|0	011111110010100010

Even though Nguyen et al. used a GA capable of solving multi-objective problems, the Non-dominated Sorting Genetic Algorithm 2 (Deb, Pratap, Agarwal, & Meyarivan, 2002) or NSGA-II, they took the sum of affordance qualities as the single objective to be maximized. They also mention how human evaluators can rate the affordances of the product, effectively capturing end-user input. However, their research did not make use of real end users; instead, they implemented a neural network which was trained by one user and mimicked input from a user. Their results suggested that product evolution was possible through the evaluation of affordance quality by end-users.

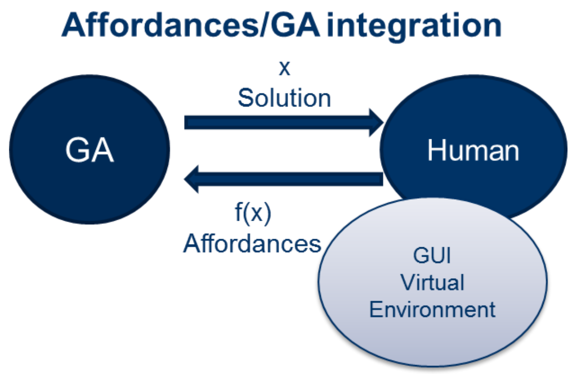


Figure 3.1 ABD/IGA integration

This dissertation builds on Nguyen et al.'s research. The ABD and GA integration (Figure 3.1) not only requires an interface where users can assess the quality of affordances but also a platform where the designer can setup the GA and the design parameters of the product. The development of this platform is described next.

3.1 Affordance Based Design Interactive Genetic Algorithm (ABIGA)

ABIGA is a web-based platform where designers can evolve products through the evaluation of affordance qualities by end-users. The web application allows designers to setup design problems which can then be made available to users. The design problems (called experiments) require the specification of the design parameters of the artifact, the minimum and maximum values that these parameters can adopt, a virtual representation of the artifact and the list of affordances that the users will evaluate. Once a design experiment has been initialized, users can access the application through a web browser, select the experiment and evaluate the affordances of the artifact in question.

ABIGA integrates ABD and a GA. The GA is used to process the evaluations from end-users to generate more design solutions. Since the evaluations are done by humans, the GA is considered to be an Interactive Genetic Algorithm (IGA).

When an experiment is started, an initial population is created by the IGA. ABIGA creates solutions from the IGA population that users interact with and evaluate (See Figure 3.2). The figure shows how the genetic algorithm and user interface work together to process solutions. The user interface takes the solutions from a database and draws the solution that is shown to the user and captures the user's evaluations. Because the solutions are saved in a database, it means that multiple users can access them at the same time. Once a population is evaluated in one generation, the IGA starts iterating. Improved design concepts are created in every generation. The stopping criterion is currently set by specifying the number of iterations the IGA should perform. The stopping criterion could also be set as an error between the population fitness average and the upper bound of that score.

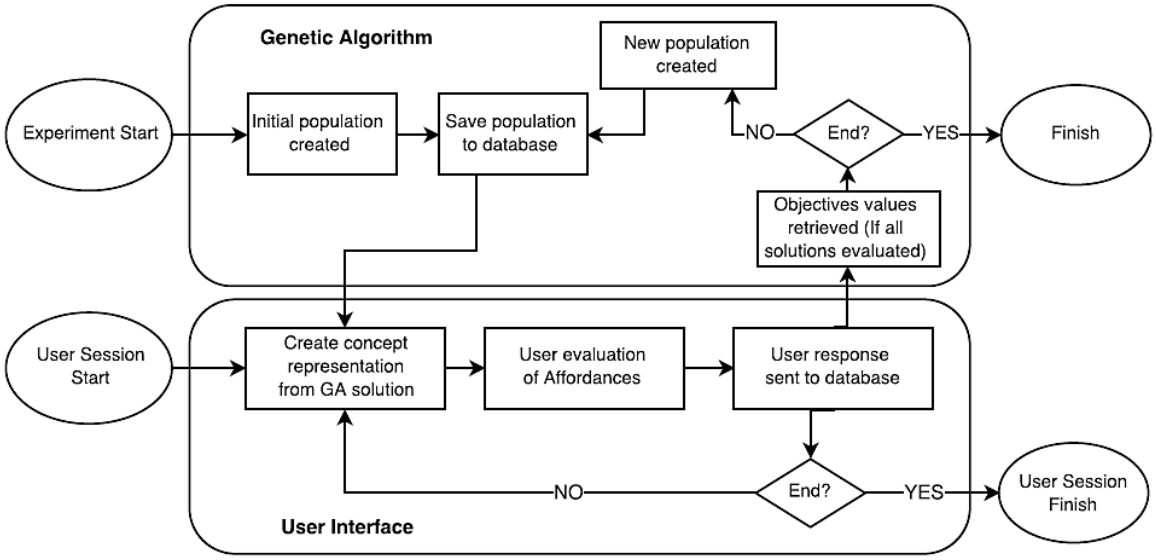


Figure 3.2 ABIGA Operation

Users can evaluate as many solutions as they wish and multiple users can evaluate solutions concurrently. ABIGA's capability of having parallel user evaluations is made possible using only one IGA; the details on how this is achieved are shown in later sections.

3.1.1 The Genetic Algorithm: Archive-based Micro Genetic Algorithm 2 (AMGA2)

A genetic algorithm, AMGA2 (Tiwari, Fadel, & Deb, 2011), is used to improve and evolve the design solutions shown to users. The GA considers the affordances as the objectives of the problem and the design parameters as the design variables. A design concept, or solution, is defined by its set of design parameter values. These parameters are used to draw each concept for users to see and evaluate its affordances.

The GA is therefore solving multi-objective optimization problems. The optimization problem is defined as the following:

$$\begin{aligned} &maximize \ \{ f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \dots, f_k(\boldsymbol{x}) \} \\ &subject\ to\ \ \boldsymbol{x} \in S \end{aligned}$$

Where $f_i(\boldsymbol{x})$ represent the affordances of the artifact. Variable vectors **x** belong to the non-empty feasible region $S \subset \mathbb{R}$.

The Archive based Micro Genetic Algorithm 2 (AMGA2) is a multi-objective evolutionary algorithm (MOEA) that borrows concepts from multiple MOEAs (Tiwari et al., 2011; Tiwari, Fadel, Koch, & Deb, 2008). This algorithm works with a small population size and keeps an external archive of good solutions found. The pseudo code of AMGA2 is shown next.

AMGA2 pseudo code (Tiwari et al., 2008):

```
Begin
   Generate initial population.
   Evaluate initial population.
   Update the archive (using the initial population).
   Repeat
      Create parent population from the archive.
```

```
        Create mating pool from the parent population and
               the archive.
        Create offspring population from the mating pool by
       crossover followed by mutation.
        Evaluate the offspring population.
        Update the archive (using the offspring popula-
tion).
     until (termination)
     Report desired number of solutions from the archive.
    End
```

AMGA2 randomly regenerates the *initial population* of solutions using Latin Hypercube sampling (Loh, 1996) along with unbiased Knuth shuffling. A slightly modified version of Differential Evolution (DE) (Kukkonen & Lampinen, 2005) is used as the *crossover* operator, which allows real variables to be used. In other words, DE allows the design variables to be encoded with real continuous values. The probability of *mutation* in AMGA2 is dynamic; it is based on the rank of the parent solutions which changes throughout the optimization run.

The size of the population can be as low as twice the number of objectives. The number of users can be large, and after a population of one generation is evaluated, a new one is generated and presented to the users. The small size of the population does mean that fewer evaluations are needed to complete a GA iteration (or generation) compared to a common GA (population size of about one hundred). The algorithm can have a reduced number of solutions in the population due to the use of an archive (see pseudo code above). An archive is created at the beginning of the optimization, once all solutions of the archive are evaluated the GA starts iterating with a small population size. The archive is updated after every generation. The solutions that are found to be better in a generation are pushed into the archive, pushing the bottom or worse solutions out of the archive.

3.1.2 ABIGA Information Structure and Database

ABIGA generates a lot of information during design experiments. This is because of the nature of IGAs that create multiple solutions that need to be evaluated. The information structure in ABIGA is designed using Entity Relationship (ER) diagrams. The ER diagrams were used to design the structure of the database that saves all of the information that is generated.

Figure 3.3 shows all of the database tables and their fields, and how they are all related to each other. The different pieces of information and their relationships are chosen based on an Affordance Based Design Ontology (Mata, Fadel, & Mocko, 2015). The ontology defines the concepts and how they are related to each other within the domain of Affordance Based Design. The database in ABIGA is coded using the MySQL language and is managed with MySQL Workbench (Oracle, 2016). Tables represent entities, or pieces of information that are saved in the database and the values of their fields, or properties, define the entity. The lines connecting the tables show how those pieces of information are related. All of the relationships in the ABIGA database are one-to-many type of relationships. For example, Experiment is related to Affordance with a one-to-many type of relationship, meaning that an Experiment can contain multiple entities of the Affordance table. In other words, there can be multiple Affordance entities related to a single Experiment.

An *Experiment* is defined as an optimization run, that is, the redesign of a product through the evaluation of affordances by users. An experiment is related to every other table in the database, either directly or indirectly. *ExpStateVars* saves the status information about each experiment, for example, the GA generation the experiment is at and whether or not people are waiting for more concepts to be generated by the GA. The *Archive* table saves the GA archive that contains a list of the best solutions found throughout the optimization run. The *Affordance* table has instances of affordances defined by their name and description. The Affordance table is related to the *Result* table, which saves the evaluations from users. A result represents the quality assessment of an affordance given by a user. Three tables describe the product being designed, namely, *Concept, DesignParameters* and *DesignParameterConstraints.* The concept is the product being designed, which is seen as a solution by the GA. Products are defined by a set of design parameters, which are all defined and saved in the *DesignParameterConstraints* table. Design parameters are defined by their name and the maximum and minimum values they can adopt during the optimization. Since the design parameters are common to all the concepts generated during the optimization run, the values that define each concept are saved in another table, *DesignParameters.*

Users evaluate the quality of the affordances of concepts; each concept has multiple affordances associated with it and the evaluations of those affordances are saved in the *Result* table. The users are asked to enter some information before they evaluate concepts; this information is saved in the *User* table.

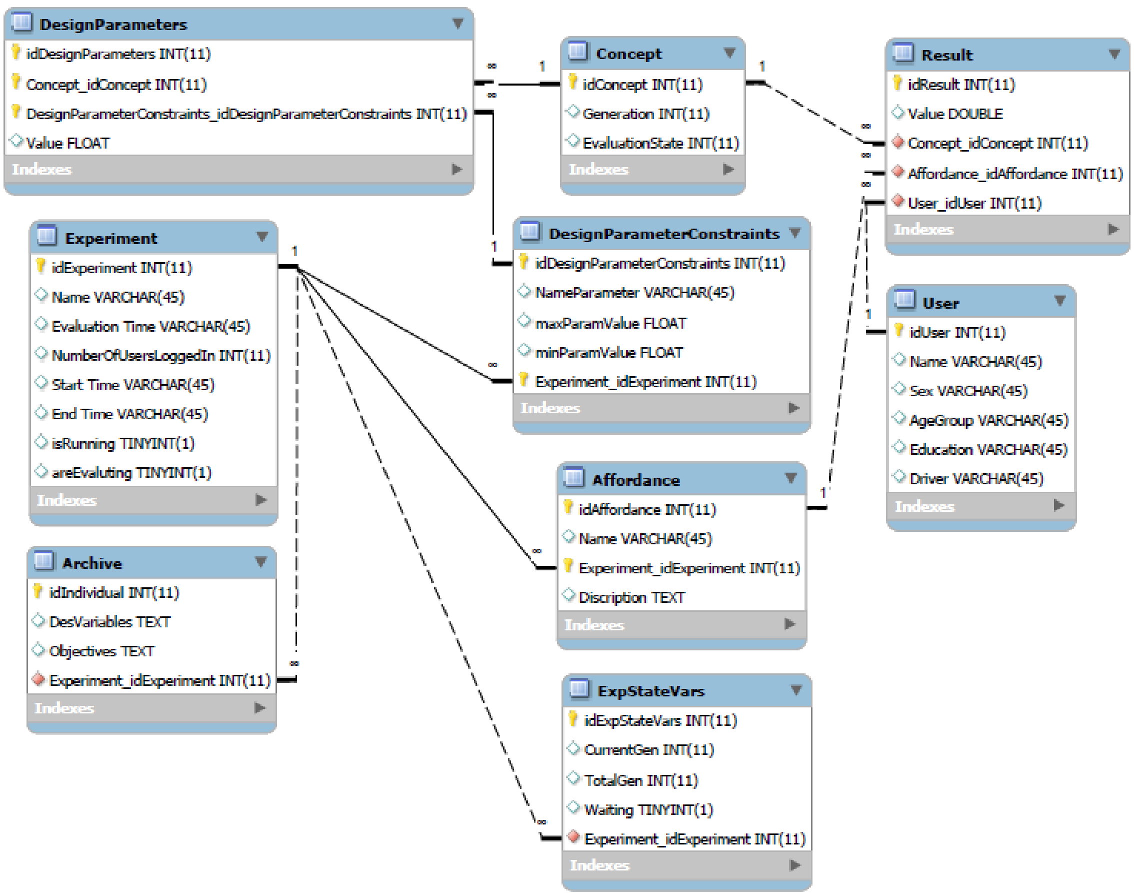


Figure 3.3 ABIGA Database Schema

All the tables in the database are related to the experiment table. Even though this increases the complexity of querying entries that are related to the experiment table, this schema eliminates redundant entries in the database.

3.1.3 ABIGA Code Architecture: The Model View Controller Structure

The model-view-controller (MVC) architecture was followed to structure the code of the web application. This structure helps separate the code that creates and handles the data from the code that presents the data (Hall, Brown, & Chaikin, 2008), taking advantage of the strengths of different technologies. For example, Java Server Pages (JSPs, combination of HTML and Java) are good at presentation, even though they allow Java code in them; their strength is in presenting data. Servlets' (Java classes that handle requests) strength however is the processing of data, even though they allow presenting it as well. Having the code structured with the MVC architecture allows for easy code modification and improvement of the application.

The use of a code structuring scheme like the MVC approach is deemed necessary when the complexity of the application is medium to high. ABIGA can be considered a medium complexity application because it integrates an IGA that constantly reads and writes to a database to present and process data from users.

As suggested previously, the basic premise of coding using the MVC structure is separating the business logic and data access layers from the presentation layers (Hall et al., 2008). In other words, it separates the code that interprets the requests (Controller) coming from users from the code that shows (View) the request results (Model) in the user interface. Implementing the MVC structure is a matter of following standard code syntaxes, the implementation steps are summarized next:

1. *Define beans to represent data.* The data that is generated should be placed in beans. Beans are Java classes that follow a standard format. There are three prop-

erties that define a Java bean. (1) A bean class should have a zero-argument constructor (the default structure of a java class). (2) A bean class should have no public instance variables (variables that can be accessed from other java classes). (3) Persistent values (global values) should be accessed through getter and setter methods (methods that read and write the values of the persistent variables).

2. *Use a Servlet to handle requests.* Servlets are Java classes that handle requests coming from users. This means that the requests should not be handled through JSP pages.
3. *Populate the beans.* The servlets trigger business logic code, or application specific code, to generate results. These results are then placed in the beans.
4. *Store the beans in the request, session, or servlet context.* The beans are saved in storages that the JSP pages (View part of MVC) can access.
5. *Forward the request to a JSP page.* The servlets transfer control to the appropriate JSP pages according to the type of request made.
6. *Extract the data from the beans.* The JSP pages (View) access the beans and their properties to present it to the users.

MVC also helps with the organization of the code because certain types of files that have the same objective can be grouped together. This is shown next in the code structure of ABIGA.

3.1.4 ABIGA Code Outline

The entire application was coded using Eclipse (Eclipse, 2014), an Integrated Development Environment (IDE) that allows the creation of applications using multiple coding languages. ABIGA is hosted on the Google App Engine (GAE) (Google, 2015a) platform, a cloud computing platform for developing and hosting web applications. The communication between the IDE and GAE is simplified with a plugin called the Google Plugin for Eclipse (Google, 2015b).

Figure 3.4 shows the project tree of ABIGA. Eclipse separates the structure of the project (a web application project) in two main folders, the source folder (src) and the war folder. The war folder follows a standard structure that simplifies the deployment of the web application. The source code files are placed in packages. There are three packages that contain multiple files in ABIGA: amga2, beans, and webAppAbiga.

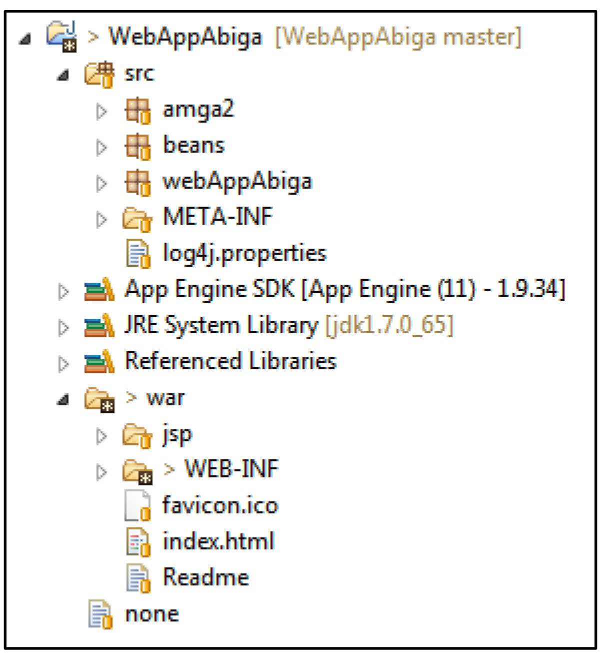


Figure 3.4 ABIGA Code Project Tree

The first package in the source folder, *amga2*, contains all the Java code for the Interactive Genetic Algorithm (IGA). Figure 3.5 shows the files that represent the genetic algorithm implemented in ABIGA. The IGA Java files are considered to be a part of the *Model* in the MVC framework, because even though instances of the genetic algorithm are not sent to users, the genetic algorithm is used to generate data objects that are sent to users.

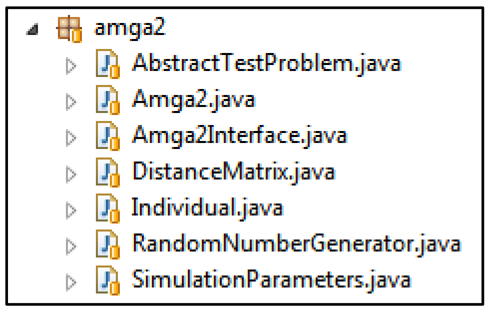


Figure 3.5 ABIGA Genetic Algorithm Java Files

The second package in the source folder is the *beans* package. The beans package contains Java files that represent objects with data. Figure 3.6 shows all the beans used by ABIGA. The beans used in ABIGA are also part of the *Model* in the MVC framework.

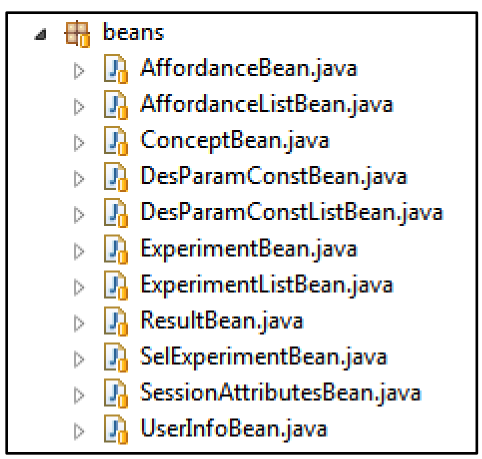


Figure 3.6 ABIGA Java Beans Files

The last package in the source folder is webAppAbiga. Most of these files are Java Servlets, which handle all requests coming from people using the application. One of these files controls all the database transactions done in the entire application (DbWrapper.java). All of the files are a part of the *Controller* in the MVC framework. Servlets manipulate the IGA and the database to generate results that are eventually shown to users.

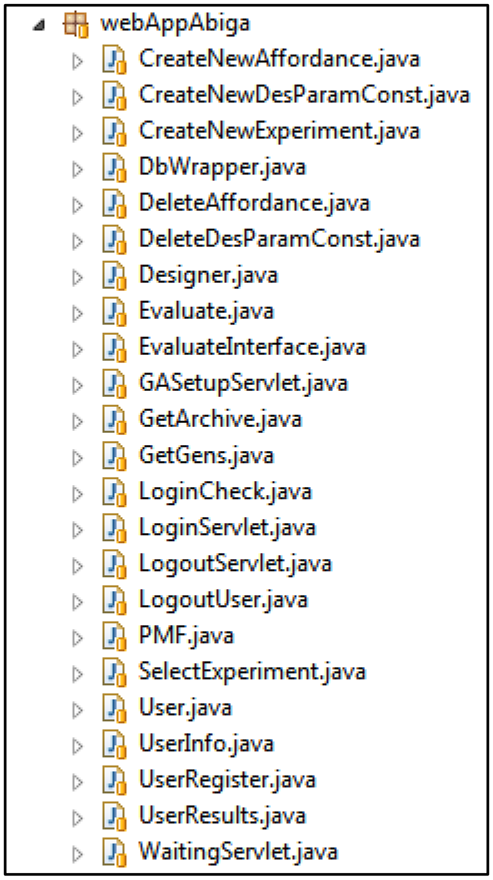


Figure 3.7 ABIGA Java Servlet and Database Files

The *View* part of the MVC framework is shown in Figure 3.8. Java Server Pages (JSP) use HTML code and JSP specific code to present data. In other words, the JSPs are the web pages users see when using the application. JavaScript language is used along JSP pages to make them dynamic.

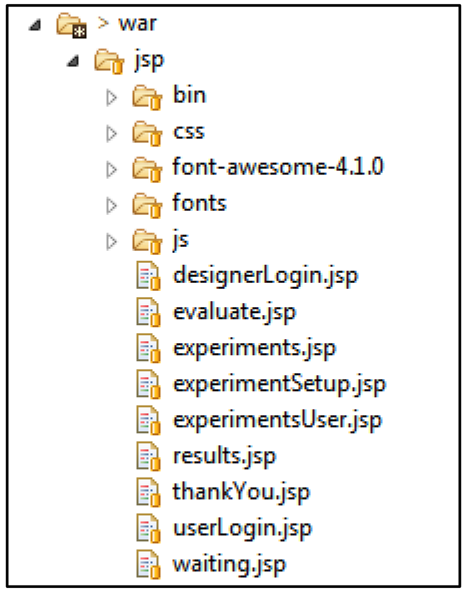


Figure 3.8 ABIGA Web Pages Files

3.1.5 Freezing IGAs to Allow Parallel User Evaluation

The integration of ABD with IGAs to evolve products requires the input of multiple users. This does not mean that a single person cannot evaluate all the solutions on their own. The results coming from multiple users would be more valuable because it would reflect the perceptions of quality from different user profiles. This adds a requirement to the development of the web application, which is allowing parallel user evaluation of product affordances. Some changes had to be done to the way the IGA works. GAs, for the most part, do solution evaluations in sequence. This can become an issue for IGAs, because it would mean that if multiple users are used as evaluators, each would have to wait too long for other users to finish evaluating their solutions.

This challenge was overcome in ABIGA using Google's Datastore technology. The Datastore is a schema less NoSQL scalable storage service. The Datastore allows the storage of data objects (such as Java classes that hold data). When a new generation of solutions is created, the GA data (state variables) is saved in the Datastore. This means the GA does not need to wait for all solutions to be evaluated once it creates them. The GA saves the solutions it creates in a database for every generation. While the GA is stored in the Datastore, all the solutions it created can be accessed from the database, which means multiple users can evaluate them at the same time. Once all solutions for a generation are evaluated, the GA is retrieved from the Datastore and uses the evaluations to continue its evolutionary operations.

3.2 ABIGA Web Application

The web application is designed with two types of access: designer and user access. Designers can create and manage experiments. To setup an experiment the designer needs to specify the affordances of the product and its design parameters. Once these have been added, the experiment can be started by specifying the IGA parameters (see Figure 3.9). Users can only select available experiments and evaluate concepts.

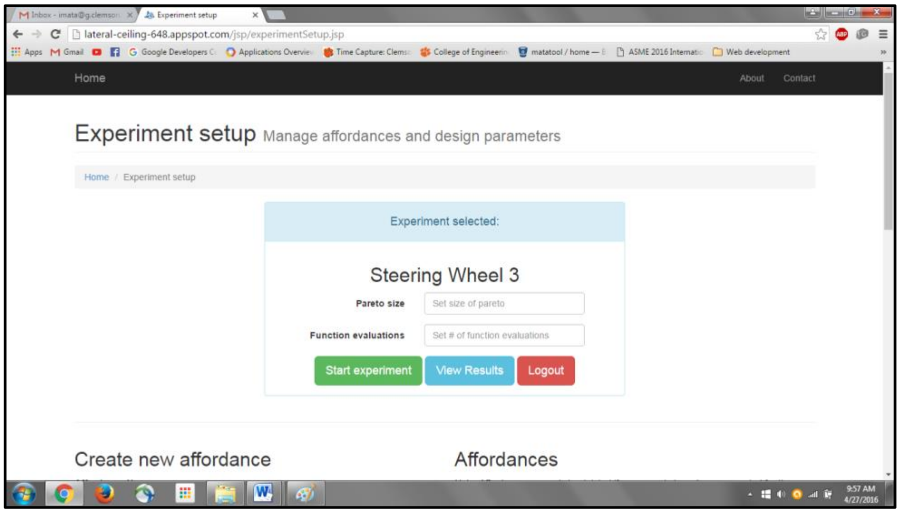


Figure 3.9 ABIGA Designer Experiment Setup

3.2.1 User Interface

Not all of the experiment setup is done through the web application. The current build of the application does not let designers create the virtual model of the product (see Figure 3.10) from the web application. The virtual model has to be programmed before the application is launched and is drawn using JavaScript code.

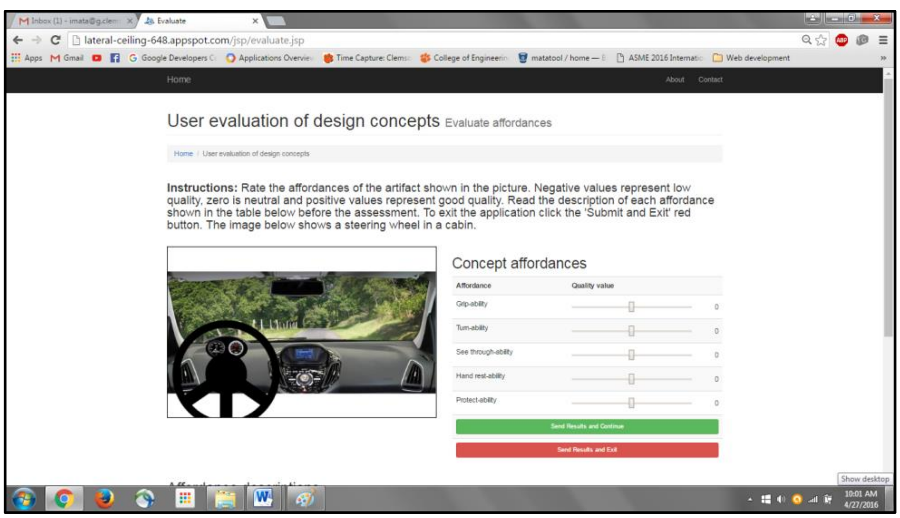


Figure 3.10 ABIGA User Evaluation Interface

The user interface shows the design concept to the user as well as lists the affordances of the concept. Sliders with a value range from -3 to +3 (in increments of 1) are given for each affordance so the user can assess the quality of those affordances. The interface of the application is based on visuals only. Some affordances require the user to touch the artifact to effectively assess their quality. Only affordances that can be assessed through a visual interface can be implemented in the current build of the application. There are technologies that could allow users to feel the solutions through haptic feedback controllers, but this limits the number of users that can be reached.

Instructions are provided to the user on how to evaluate the affordances of the product and what their meaning related to quality is. The center of the page is where users spend most of their time. The left section shows a 2-dimensional drawing of the product, which changes according to the solution extracted from the database. The right section has a list of the affordances of the product. Each affordance has a slider where users specify the quality of that affordance for the product shown on the left. There are two buttons below the affordance list in the page. The green button sends the results for the current solution to the database and loads a new solution for users to evaluate. The red button sends the solution's evaluations to the database and logs the user out of the experiment, which means they do not get any more solutions to evaluate. The bottom of the page (not shown in the figure) contains the descriptions of all the affordances of the product. Users are instructed to read these before they begin their evaluations and can always refer back to them in case they need to.

The evaluating page is not the only page users get to interact with. There are other pages users see before they get to the evaluating page. For example, users are required to provide some information (age, sex, education level, name) in a login page. This information can be used to aggregate the results of the evaluations and target the design based on those parameters. Users have to select the experiment they wish to be a part of in an experiment selection page and finally, they're taken to a thank you page when finished.

3.3 Setting up Experiments in ABIGA

The steps to set up experiments in ABIGA are defined in Figure 3.11 and are explained next.

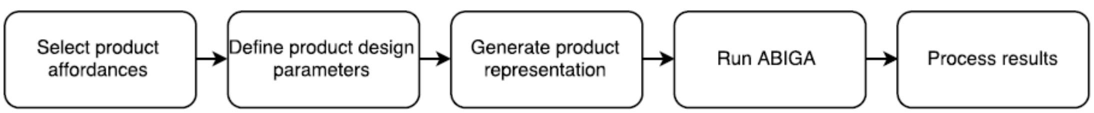


Figure 3.11 ABIGA Experiment Set Up Steps

3.3.1 Select Product Affordances

The designer is expected to use ABIGA during the embodiment stage of the design process (use of ABIGA in other stages will be discussed in later chapters). This means that the designer has a list of affordances that he believes the users will perceive; users of course may perceive more. The list of affordances includes all types of affordances, namely, AUAs, AAAs and AEAs. However, the user is not expected to be able to assess the quality of all types of affordances. This is because some affordances describe the interaction between entities that the user does not see. For example, the user could not assess the quality of the AAA that describes the power transmission between a piston and a crankshaft in a reciprocating internal combustion engine; not only because the user cannot interact with these components, but also because this can be quantified with the physics of the system. This does not mean that the user cannot assess the quality of all AAAs or AEAs. There could be some AAAs or AEAs that can be assessed by users if at least one of the interacting entities can be perceived by the user. An example of this would be the *exposure-ability* of a camera, which is an AEA that describes how much light enters the camera to be processed by its sensors. The exposure-ability of a camera depends on the size of the lens of the camera, a component that the user is able to perceive.

The set of affordances that the designer can include in an ABIGA experiment do not represent the entire set of affordances needed to design the artifact. Most of the affordances chosen by the designer are expected to be AUAs, simply because these affordances describe the interactions between the user and the artifact and might have an effect on the shape of the artifact. Furthermore, the designer would consciously select the affordances he may want the users to think of when looking at the designed object, and the designer may want the users to affect the design in a specific way if he believes that a certain shape or feature would elicit a certain response from the users.

ABIGA has a designer interface where the affordances can be added. The interface is part of the Experiment Set Up page and is shown in Figure 3.12. Besides entering the name of the affordance, a description of the affordance is also needed. *It's important that the description of the affordances do not give any clues on how the product needs to* *be used*. This is to not predispose users to focus on specific aspects of the product which would influence the affordance/design parameter relationships found in the results.

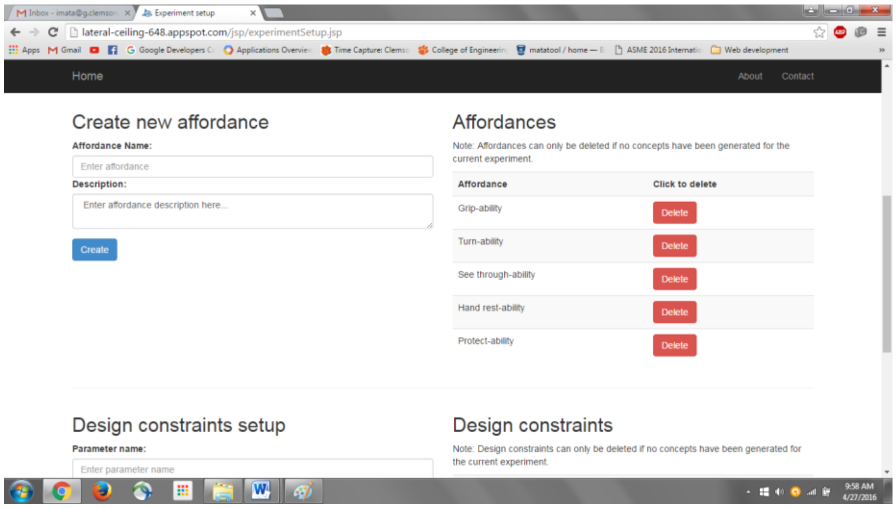


Figure 3.12 ABIGA Affordance Specification Interface

3.3.2 Define Product Design Parameters

The set of design parameters need to be chosen based on the type of product representation used in the experiment. ABIGA currently only supports two-dimensional product representations. This means that the design parameters should be enough to define the shape of the product in two dimensions. The set of design parameters should also be related to the same product topology. This is because ABIGA currently does not support the change of product topology during the optimization run (possibilities on how to make this happen are discussed in later chapters).

The modular aspect of ABIGA's architecture allows for easy upgrades. Three-dimensional product representation capabilities can be added in future iterations of ABI-

GA. This would affect the choice of design parameters that are needed because of the extra dimension.

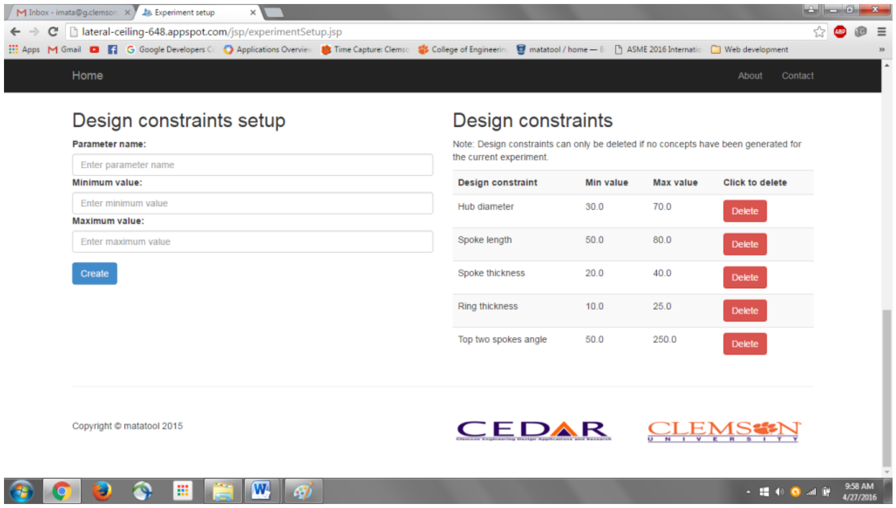


Figure 3.13 ABIGA Design Parameter Specification Interface

Figure 3.13 shows the interface where designers specify the different product design parameters and the range of values each variable can adopt during the optimization run.

3.3.3 Generate Product Representation

This step goes hand in hand with the previous step. This is because the design parameters are used to generate the product representation of the product. As mentioned earlier, ABIGA currently supports two-dimensional product representations. To generate the two-dimensional product representation, a JavaScript file needs to be created for every product. The order in which the design parameters are used in the JavaScript file should be in accordance with the order in which the design parameters were entered in the previous step.

It may seem counterintuitive to use two-dimensional representations of products, but research has shown that user perceptions of basic forms are consistent with user perceptions of their full models (Orbay et al., 2015). This means two-dimensional representations of products do not affect user perceptions of simplified forms.

Future iterations of ABIGA can integrate the specification of the design parameters with the creation of the product representation steps. This can be achieved by adding a drawing interface that automatically pulls the design parameters to be used in the optimization run and product representation tool.

3.3.4 Running ABIGA and Processing Results

The ABIGA platform can be accessed online. Users are given the URL of the application and can access the experiments that the designer has started from a list. The best solutions found by ABIGA can be extracted from a database. Currently there is no automated way to perform data analysis, which is why the affordance/design parameter relationships are checked using a separate statistics analysis software. More details about these steps are given in the following chapter.

3.4 Summary

This chapter provided a generalized view on how the integration between Interactive Genetic Algorithms (IGAs) and Affordance Based Design (ABD) works as well as how ABIGA was developed. The IGA/ABD integration allows end-user input to be captured in the form of affordance quality assessments. The IGA solutions are a set of design variables that represent a product, defined by its set of design parameters. The IGA objectives are the set of affordances of the product. The evolutionary operators of the IGA and the input from end-users are used to find solutions with high affordance quality scores. This can be seen as a multi-objective optimization problem where the affordance quality values are being maximized.

ABIGA is an online design tool that evolves products by capturing user input. Designer can setup design experiments by specifying the affordances of a product and its design parameters. Users can access experiments and assess the quality of product affordances. The development of ABIGA involves multiple programming languages: Java, MySQL, JSP/HTML, JavaScript/JQuery and XML. The application is deployed through the Google App Engine (GAE), which provides multiple services besides hosting the application (Datastore, Database).

ABIGA's code is organized based on the Model-View-Controller (MVC) Architecture. This framework separates the code that is used to generate data (Model), the code that is used to show the data (View) and the code that is used to handle all requests coming from the application users (Controller). The main advantage of using the MVC framework is that it simplifies the way modifications are done to the application, as well as the way features are added to it.

ABIGA solves an important issue regarding real user evaluations. Due to the nature of GAs, the solutions that are generated are evaluated in sequence. This becomes a problem when real users are implemented because it limits the number of users that can perform evaluations to one at a time. ABIGA allows concurrent user evaluation, significantly reducing the time the IGA takes to iterate. This is done by saving the solutions created by the IGA in a database and saving an image of the GA itself. The solutions can be distributed to multiple users at the same time and once they are all evaluated the IGA can be restored and fed the input from the users to continue the optimization operations.

The following chapter shows how the ABIGA application was used to evolve a steering wheel and how the results can be used to link the affordances of a product with its architecture.

Chapter 4

4 Proof of Concept: Redesign of a Steering Wheel

A steering wheel and a camera are used to test if the ABD/IGA integration could evolve products through the quality evaluation of their affordances by end-users. This chapter will show the results of the steering wheel experiments, the results of the camera experiments are given in Chapter Five. Why a steering wheel? As mentioned in Chapter One, affordances can describe the interactions between the product and the user. This suggests that the more the user interacts with a product, the more AUAs (artifact-user affordances) he or she may perceive that the product could have. This of course depends on the type of user. For example, a car engine may have just a few AUAs for the end-user because physical interaction is not expected during operation. However, if the type of user is a servicing mechanic, then multiple AUAs would be perceived. End-users get to constantly interact with a steering wheel when driving a car, which means that more affordances may be perceived and are available to work with. Furthermore, Nguyen et al. (Nguyen et al., 2012) chose a steering wheel as their example to show how the ABD/GA integration can be used to evolve products. As mentioned previously, Nguyen et al. used neural networks for their affordance quality assessments, choosing the same product will help validate the integration with real end-user evaluations.

4.1 Experimental Test Method

The objective of the experiments is to test product evolution by having an IGA optimize the design parameters of a steering wheel where the objectives of the optimization are its affordance qualities as perceived by the users. A total of six steering wheel experiments were performed. Three of those experiments were performed using real end-users. Three is considered to be the minimum number of experiments for the results to have a statistical significance. The other three experiments were performed using a random number generator (RNG) as the evaluator of quality affordance. This is to prove that the results cannot be obtained through input randomness.

All experiments were done under the same conditions to allow comparison between the results. The experiment variables considered are shown next:

1. *Affordances and design parameters of the steering wheel.* The same affordances and design parameters were used in all experiments. The affordances and design parameters are shown later in this chapter.
2. *Number of users.* In principle, any number of users can be involved in ABIGA experiments. However, the number of users in each experiment is kept constant to consistently compare the results of all experiments. A total of six users with similar backgrounds were asked to evaluate the affordances of the steering wheel.

3. *IGA parameters.* The parameters of the IGA were kept constant. The parameters include the size of the archive (specific to the type of GA used, AMGA2), the size of the working population, crossover and mutation rates and number of function evaluations.

4.1.1 The Steering Wheel

The steering wheel is modelled with five design parameters and five affordances specified by the author. As pointed out in the previous chapter, the chosen affordances are not the entire set of affordances needed to design a steering wheel. Other affordances may exist, but may not be perceived by users, especially artifact-artifact affordances (AAAs). For example, the affordances related to interactions between the steering shaft (if there is one at all) and the rack and pinion system. If that interaction is poor, noise may be generated, and the negative AAA may be identified by a trained person. However, since the objective of the work is to help in the design process, *the designer needs to select a set of what he or she thinks are relevant affordances that users can evaluate and that will have an impact on the form of the product*. The steering wheel has five design parameters (see Table 4.1). Pixels were used as the units for the design parameters (with the exception of the angle parameter) as a proof of concept; in industry applications, designers would of course use appropriate units. The value range of each parameter is chosen to match the size restrictions of the dashboard image used as the background of the steering wheel. These values can be converted to real size values when compared to the size of the dashboard used as a background. Pixels can first be converted to millimeters

(1 mm ≈ 3.78 pixels) and then this value can be scaled back to normal size. For example, the dashboard width is 640 pixels (169.3 mm) in Figure 3.10. If a dashboard's width is 1300 mm, then the scaling factor needed would be approximately 7.68 ($scale\ factor = \frac{real\ size}{image\ size}$). For the steering wheel used in the experiments, the overall diameter varies from 143 mm to 260 mm.

Table 4.1 Steering Wheel Design Parameters

Design parameter	**Minimum value**	**Maximum value**
Hub diameter (HD)	30 pixels (61 mm)	70 pixels(142 mm)
Spoke length (SL)	50 pixels (102 mm)	80 pixels (163 mm)
Spoke thickness (ST)	20 pixels (41 mm)	40 pixels (82 mm)
Ring thickness (WT)	10 pixels (20.5 mm)	25 pixels (51 mm)
Top two spokes angle (SA)	50 degrees	250 degrees

In this experiment, only the shape of the steering wheel varies. The topology stays the same, which means that the number of spokes remains constant (three spokes). Changing the topology could be possible by creating features that can be added to the product controlled by a Boolean variable (discussed later in the chapter). However, the challenge with this approach is that new design parameters are introduced with each pre-determined feature, increasing the total number of design parameters that the GA has to work with. Changing the topology of products in the middle of the optimization run is important and can be addressed in future work; but it is outside of the scope of this research.

The shape of the steering wheel is shown in Figure 4.1. The name of the design parameters are shown in Table 4.1.

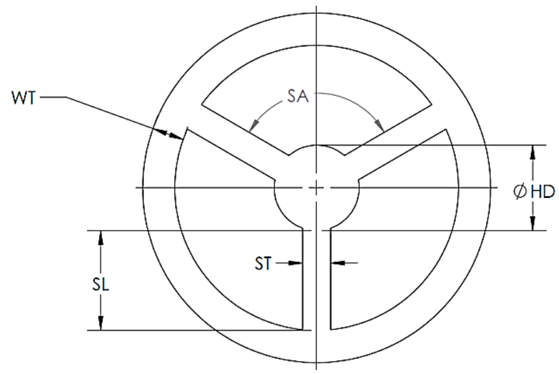


Figure 4.1 Steering Wheel 2D Drawing

The affordances of the steering wheel were the same in all experiments. There are five affordances in total in this experiment, the affordances and their descriptions as shown to the users can be seen in Table 4.2.

Table 4.2 Steering Wheel Affordances and Descriptions

Affordance	Description
Grip-ability	Gripping interaction between the hands of the user and the steering wheel.
Turn-ability	Interaction between the hands of the user and the steering wheel where the user rotates the steering wheel to turn the car.
See through-ability	Visual interaction between the user and the steering wheel that allows the user to see through the steering wheel. This could affect the visibility of the dashboard gauges and or the street.
Hand rest-ability	Interaction between the hands of the user and the steering wheel that allows users to rest their hands on any part of the steering wheel.
Protect-ability	Interaction between the user and the steering wheel where the steering wheel protects the user in the event of a crash.

4.1.2 Users

The users were given the web application's URL and were instructed to choose the steering wheel experiment. There was no interaction with any user when evaluating concepts. Each of the three experiments was evaluated by different sets of six users. All users were mechanical engineering graduate students. Users were asked if they were drivers, only one of the users in the first experiment reported that s/he was not a driver.

4.1.3 Genetic Algorithm Parameters

The GA (AMGA2) parameters that can be changed in ABIGA are the size of the archive and the number of evaluations performed. The latter is equivalent to the number of generations during the optimization run because the size of the working population is known (twice the number of objectives). All three experiments had an archive size of fifty and were run for a total of fifteen generations. Fifteen generations were chosen based on the experiments performed by Nguyen et al. (Nguyen et al., 2012). Their results showed the population fitness reaching the upper limit at around twelve generations. Other simulation parameters such as the crossover probability (p=1.0) and the mutation probability (p=1/N, where N is the number of design variables) were not changed between experiments. Note that our objective is not to fine tune the GA to get the best and most efficient results possible, rather to show that with a standard set of parameters, and no tuning, the GA can evolve the solutions using user feedback, and obtain better solutions.

4.2 Experiment Results

ABIGA stores a lot of information for each design experiment. Every concept generated by the GA is stored in a database. This includes the design parameter values and the affordance evaluations of each solution. All of this information can be queried from the database for analysis while the experiment is running or after it is completed. Appendix A shows MySQL code used to extract data from the database. The data extracted from the database is graphed using Minitab 16 (Minitab, 2016).

4.2.1 Product Evolution

To check if the solutions in the IGA improve across the different generations, the fitness of the entire population can be tracked. Since there are multiple objectives (affordances), the overall fitness of a solution can be reduced to the sum of all its objectives. This is done only for easy visualization of the evolution of the product. This is not how the IGA operates. The IGA is a multi-objective algorithm that uses the rankings of each objective (affordances) to evolve solutions towards better ranked solutions that eventually become non-dominated, and possibly Pareto. Details on how the GA works can be found in (Tiwari et al., 2011, 2008). The graph does show however the trend towards solutions that are perceived to be better by the users, considering all of the affordances. Figure 4.2 shows the averages of solutions' fitness values across all generations for each experiment along with a 95% confidence interval for the mean. Though possible, this does not mean that the solutions are converging in the solution space. The solution are improving based on the perceptions of users, meaning the population fitness is reaching the upper limit.

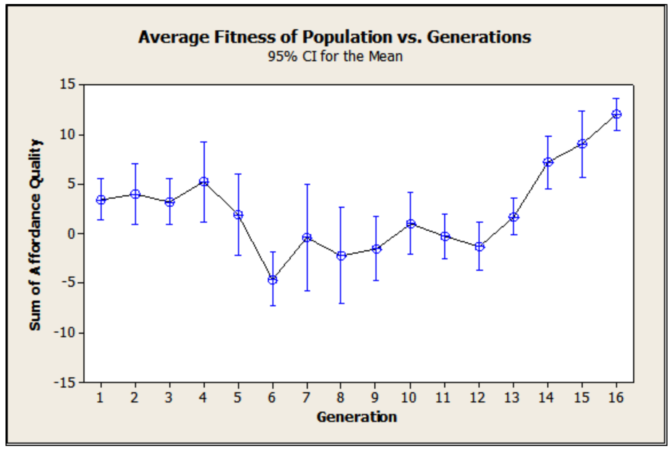
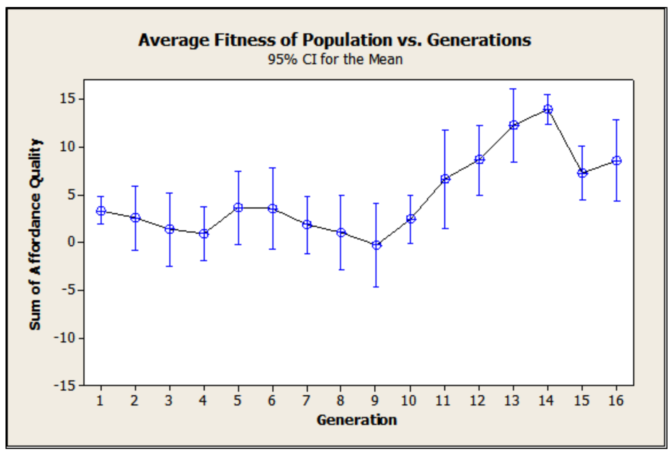
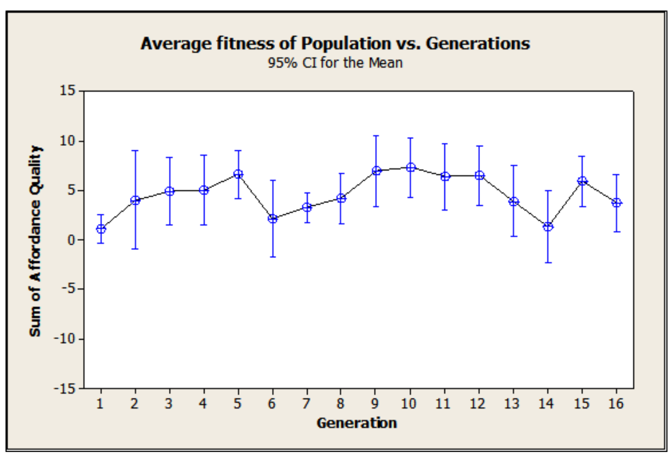
<table>
<tr><td>Experiment 1</td><td>
</td></tr>
<tr><td>Experiment 2</td><td>
</td></tr>
<tr><td>Experiment 3</td><td>
</td></tr>
</table>

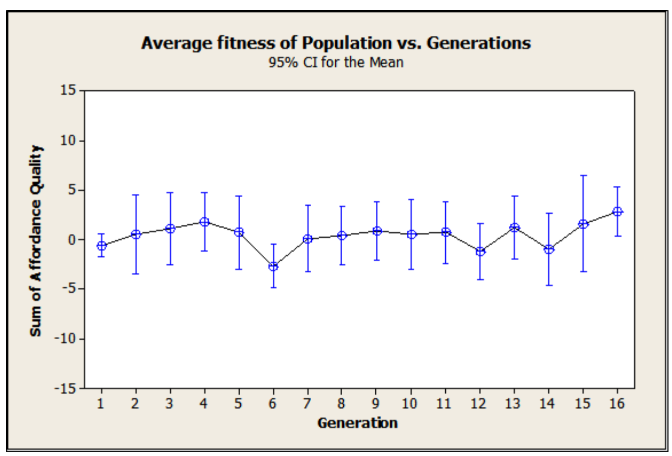
Figure 4.2 Steering Wheel Evolution: Real User Results

The graphs show sixteen generations because the first generation is the initial population, which also has to be evaluated by users. The maximum value possible for the overall fitness is fifteen, as there were five affordances, each of which could have a maximum quality value of three. The discussion of these results is presented after the description of the results of the same experiments ran with a random number generator.

4.2.2 Random Number Generator as Input

To prove that the steering wheel cannot evolve as a product of chance, experiments were done using a Random Number Generator (RNG) as the affordance quality evaluating function. A total of three experiments were performed using the RNG input. The evolution results are shown in Figure 4.3.

Experiment 1



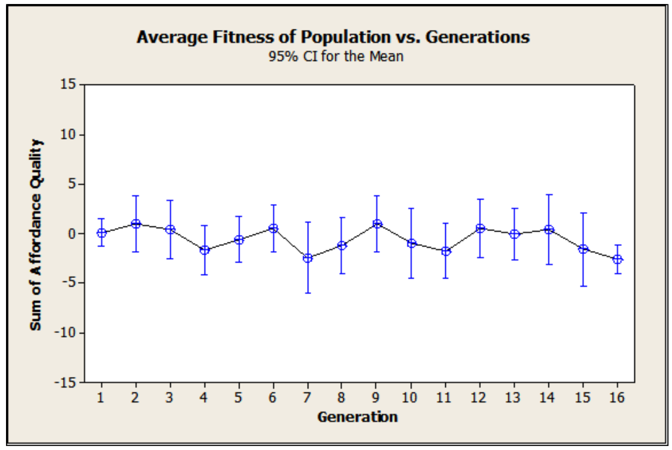
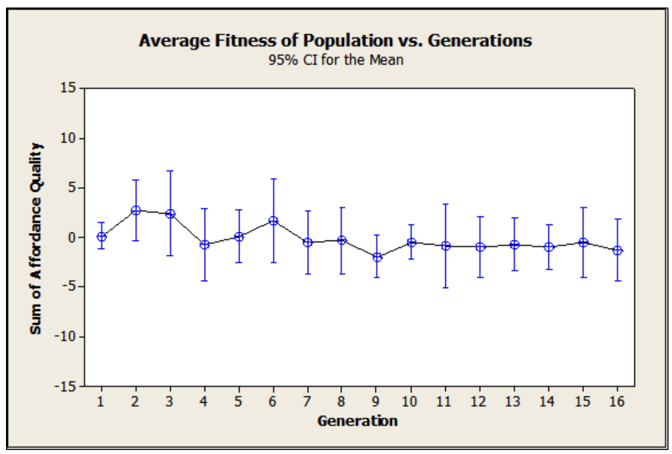
Experiment 2	
Experiment 3	

Figure 4.3 Steering Wheel Evolution Results: RNG Input

4.2.3 Discussion on Product Evolution

4.2.3.1 Real User Experiments

The results in Figure 4.2 show that the IGA evolves the steering wheel shape towards solutions that are perceived (by users) to be better than the solutions in previous generations. These results are in agreement with the results obtained by Nguyen et al. (Nguyen et al., 2012), which show how a GA evolved the shape of a steering wheel using an Artificial Neural Network (ANN) trained by one user as the affordance quality evaluator.

A steady increase can be seen in the experiments. For example, Experiment 1 results show such an increase at the twelfth IGA generation. Experiment 2 shows the increase at the ninth generation. Experiment 3 shows that increase at the tenth generation. Though there is a decrease in population fitness in Experiment 2 after the steady increase, the results are still valid since a maximum is reached. It is not expected that all the solutions in the best generation will reach the maximum score (max. score = 15). Moreover, the IGA saves the best solutions in an archive, meaning it doesn't matter if the last generation is not the best of them all, because the archive will always keep track of the best solutions found across all generations.

There could be a couple reasons why the population fitness can decrease after reaching an apparent maximum. One reason is that the IGA is always trying to explore as much of the solution space as possible by using one of its key evolution operations: *mutation*. Mutation basically transforms a solution into a new solution, sending it to other areas within the solution space to avoid getting trapped in local minima. Another reason is the introduction of evaluators late in the optimization run, which was the case on experiment two and three. Even though the population's fitness average decreased on generation fifteen for experiment two, the graph shows that this average increased by generation sixteen. This suggests that the average would eventually reach the maximum once again.

Even though experiment three shows areas of constant fitness increase, it did not reach the fitness values that were seen in experiments one and two. A possible reason for this is the way that users evaluated the experiment. In experiments one and two, all users were notified to assess the affordances of an experiment at the same time. This means that there were inputs from multiple users at every generation. In experiment three, users were not all asked to evaluate affordances at the same time. In other words, the chance of having the input from multiple users in every generation was significantly reduced. This is important because the more users there are in the initial population the more diverse the inputs will be. Having only one or a few users assess most of the initial population would make it harder for the IGA to learn the input profile of all users. This suggests that early input diversity helps the IGA reach optimal solutions in fewer generations. To test this, experiment 3 was run past the fifteen generations set for the initial experiments.

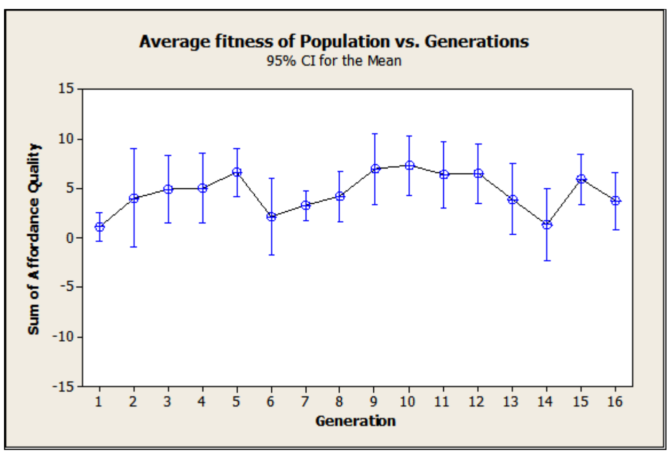


Figure 4.4 Steering Wheel Evolution Experiment 3 (repeat)

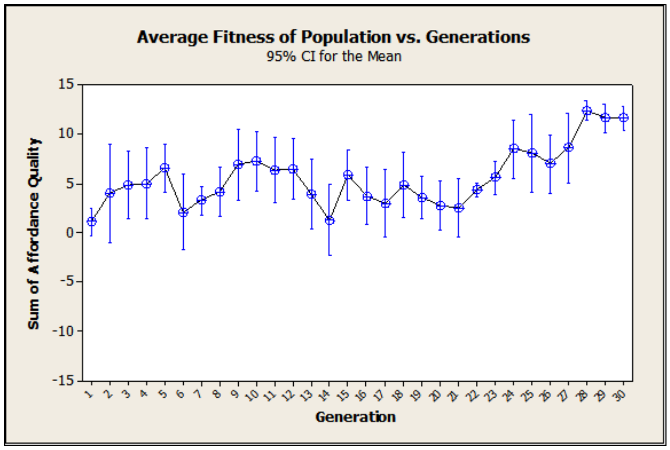


Figure 4.5 Steering Wheel Evolution Experiment 3 Extended

Figure 4.4 is shown again for comparison purposes. Figure 4.5 shows the results of the same experiment past the sixteenth generation. Three more users were involved in the evaluation of solutions past the sixteenth generation. The evolution of the steering wheel eventually reaches the same levels obtained in experiments one and two. This suggests that if multiple users are involved in the evaluation of affordances, the IGA will reach optimal solutions faster when users' input is combined early in the experiment.

The results of this extended experiment also show that even if more users are introduced during the optimization run, the population fitness grows toward "good" solutions (i.e., generations with high fitness values). The IGA in this case takes more generations to learn the profiles of the users.

4.2.3.2 Random Number Generator Experiments

Figure 4.3 shows the results of three experiments where the user input was replaced with a Random Number Generator (RNG). None of the experiments reach the population fitness values seen in the experiments with real end-users. This can be appreciated in the way that the fitness score of the population never improves in a steady manner for more than two generations as seen in the real end-user experiments. The behavior can be attributed to the fact that the IGA cannot identify any patterns from the randomized input. These results suggest that products cannot evolve toward "good" solutions out of chance; which validates the results obtained with real end-users.

The fact that products evolved toward better perceived solutions in the experiments with real end-users has two major implications:

1. Real users can be used to assess the quality of affordances and effectively evolve products toward better perceived solutions.
2. The ABD/IGA integration can optimize the shape of products

4.2.3.3 User Input Combination to Evolve Products

One of the key limitations that Nguyen et al.'s work faced was the fact that real users did not evaluate the solutions found by the GA. ABIGA's steering wheel results prove that it is possible to use the input from multiple users to evolve products. The IGA in ABIGA is essentially combining the input from the evaluators. This is because of the way the IGA creates new solutions using *Crossover* as one of its evolutionary operations. However, it cannot be concluded that the optimized results reflect the preferences of all users. This is because of Arrow's Impossibility Theorem (Hazelrigg, 1996), and due to the fact that solutions are only evaluated once (The probability of the IGA creating the same solution twice is small). Nonetheless, the results can be understood as combinations of preferred features. For example, User-A may have considered the *seeThrough-ability* of Concept-*m* as being of high quality (quality score > 1), but considered its *grip-ability* as low quality (quality score < -1). On the other hand, User-B considered the *seeThrough-ability* of Concept-*n* as low quality but perceived the *grip-ability* as high quality. The IGA in this case (if both concepts are taken to the mating pool and chosen for crossover) will most likely create another solution that includes the high quality features from both solutions, creating Concept-o, with a high quality *seeThrough-ability* and *grip-ability* if the parameters enabling such perceptions are different.

4.2.3.4 Optimization of Product Shape

As shown in Chapter 3, the IGA in ABIGA is solving multi-objective problems. The results obtained are optimized solutions based on the inputs from end-users. Since the optimization variables are design parameters that define the shape of the product, it can be said that ABIGA optimizes product shape. The results are valuable not only because relationships between design parameters and input from end-users can be obtained from the evaluations (discussed later in the chapter), but because the results consider the effect that different affordances have with each other with respect to product shape. For example, when users evaluate the *seeThrough-ability* they might look at the angle between the top two spokes but also notice how this feature also affects *handRest-ability*. This is because users are not evaluating individual features of the product, but the product as a whole.

The current build of ABIGA only supports product shape optimization. There are no changes in the topology, for example, the number of spokes is not a variable in the steering wheel experiments. However, this doesn't mean that ABIGA cannot handle topology changes in design problems. There are two scenarios in which topology changes can be implemented. The first scenario is by using predetermined features that can be added to the product. For example, the steering wheel can have bumps on the ring that improve its *grip-ability;* this could be considered a feature with its own set of design parameters that can be present or not in the steering wheels. The only drawback to this approach is that the designer needs to know the features that can be in the product beforehand. The features could also be recommended by the users themselves so that the designers can add such features in future experiments. Nonetheless, there are benefits in separating the basic shape of the product from extra features that it might have: besides optimizing shape, ABIGA could also be used to optimize product configurations.

4.2.4 Relationships between Affordance Quality and Design Parameters

The existence of relationships between affordances and design parameters is suggested by one of the properties of affordances: *form dependency*. This property says that the affordances in a product are dependent upon the shape or geometry or physical characteristics of objects. For example, the form of a big box doesn't offer hand grip-ability, but if a handle is added to the sides of the box, therefore changing the geometry of the box, grip-ability is now possible due to that change. Having relationships between affordance qualities (as perceived by users) with the design parameters of the solutions would mean that the designer could select design parameter values that consistently get positive ratings by users to target specific product affordances.

The affordance quality input from users is categorical, that is, the variables are discrete, ranging from -3 to +3. Other scales can be used, this linear scale is used to match the scale used by Nguyen et al. (Nguyen et al., 2012) in their experiments. Though the effect that the scale might have on the evolution results is not expected to yield significant differences, such effect is not studied in this research. Users were instructed that negative values meant a bad affordance quality, zero meant neutral and that positive values meant the product had a good affordance quality. Unlike the affordance variables, the design parameter variables are continuous. Due to this discrepancy, linear regression techniques cannot be implemented between these variables to determine if there are any relationships between them. Instead, logistic regression is the recommended method to test relationships between categorical and continuous data.

The affordance variables are therefore further categorized in the form of a binary response. Table 4.3 summarizes how this categorization is done. The zero-valued responses are not used in the analysis and may be used in the future to add dummy affordances and let the users define what other affordances they perceive. This topic is discussed in the future work section in the last chapter.

Table 4.3 Binary Categorization of User Response

7-Point Likert scale	-3	-2	-1	0	1	2	3
Binary response	Not good (0)				Good (1)		

The reason for making the affordance variables binary is because designers are interested in what users consider to be "good" solutions. If a relationship between a design variable and an affordance exists, the binary logistic regression could tell us the design parameter values that are more likely to be perceived as positive affordances by the users.

The binary logistic regression analysis was done using Minitab 16 (Minitab, 2016). Table 4.4, Table 4.5 and Table 4.6 show the results of design parameters tested against all affordances for all real-user experiments. The P-value is shown in the table for each pair of design parameter and affordance. If the P-value is less than 0.05 then there is evidence that the design parameter contributes to the prediction of the affordance quality outcome. The significant values are highlighted in the tables.

Table 4.4 Binary Logistic Regression P-value Results Experiment 1

	HubDiameter	**SpokeLength**	**SpokeThickness**	**RingThickness**	**TopTwoSpokesAngle**
Grip-ability	0.071	0.214	0.981	0.404	0.042
Turn-ability	0.086	0.023	0.475	0.173	0.001
SeeThrough-ability	0.009	0.202	0.015	0.018	0.000
HandRest-ability	0.196	0.525	0.045	0.194	0.000
Protect-ability	0.956	0.065	0.011	0.015	0.685

Table 4.5 Binary Logistic Regression P-value Results Experiment 2

	HubDiameter	**SpokeLength**	**SpokeThickness**	**RingThickness**	**TopTwoSpokesAngle**
Grip-ability	0.055	0.885	0.980	0.317	0.021
Turn-ability	0.002	0.985	0.049	0.120	0.009
SeeThrough-ability	0.001	0.000	0.030	0.002	0.000
HandRest-ability	0.012	0.152	0.638	0.653	0.000
Protect-ability	0.701	0.000	0.003	0.000	0.968

Table 4.6 Binary Logistic Regression P-value Results Experiment 3

	HubDiameter	SpokeLength	SpokeThickness	RingThickness	TopTwoSpokesAngle
Grip-ability	0.452	0.059	0.019	0.004	0.137
Turn-ability	0.651	0.088	0.298	0.802	0.001
SeeThrough-ability	0.362	0.003	0.088	0.060	0.000
HandRest-ability	0.159	0.973	0.682	0.048	0.000
Protect-ability	0.001	0.029	0.017	0.716	0.094

The relationships between Affordances and Design Parameters were also determined for the experiments with random inputs. Table 4.7, Table 4.8 and Table 4.9 show the relationships found.

Table 4.7 Binary Logistic Regression P-value Results Experiment RNG-1

	HubDiameter	SpokeLength	SpokeThickness	RingThickness	TopTwoSpokesAngle
Grip-ability	0.953	0.887	0.846	0.288	0.480
Turn-ability	0.721	0.542	0.062	0.163	0.210
SeeThrough-ability	0.089	0.441	0.692	0.616	0.033
HandRest-ability	0.503	0.832	0.888	0.136	0.889
Protect-ability	0.112	0.209	0.722	0.705	0.101

Table 4.8 Binary Logistic Regression P-value Results Experiment RNG-2

	HubDiameter	SpokeLength	SpokeThickness	RingThickness	TopTwoSpokesAngle
Grip-ability	0.652	0.575	0.462	0.081	0.250
Turn-ability	0.767	0.901	0.427	0.636	0.784
SeeThrough-ability	0.510	0.035	0.825	0.083	0.314
HandRest-ability	0.428	0.211	0.235	0.459	0.187
Protect-ability	0.921	0.614	0.052	0.572	0.884

Table 4.9 Binary Logistic Regression P-value Results Experiment RNG-3

	HubDiameter	**SpokeLength**	**SpokeThickness**	**RingThickness**	**TopTwoSpokesAngle**
Grip-ability	0.819	0.052	0.015	0.556	0.977
Turn-ability	0.500	0.180	0.547	0.533	0.281
SeeThrough-ability	0.187	0.174	0.320	0.337	0.542
HandRest-ability	0.403	0.709	0.198	0.756	0.674
Protect-ability	0.907	0.230	0.463	0.293	0.864

4.2.5 Discussion on Affordance vs. Design Parameters Relationships

There is no single solution that will be preferred by all users (Arrow's impossibility theorem (Hazelrigg, 1996)), since there cannot be a solution that aggregates everyone's preferences. Therefore, it does not make sense for the designer to expect to use a single set of design parameter values obtained from ABIGA in their design. After all, we wouldn't expect that a quarter degree in the angle between the top two spokes of a steering wheel would make a significant difference to the perception of quality of a specific affordance from users. It would make more sense if designers had ranges of values for each design parameter that they could work with so that whatever values were chosen would elicit good quality perceptions for various affordances from the users. It turns out this is possible if the responses from the users show that there are relationships between the design parameters and the affordances of the product in question.

As shown earlier, using logistic regression techniques, relationships between design parameters and affordances were found. These results suggest that designers can target specific affordances by changing the values of specific design parameters.

One of the many relationships found was between *Turn-ability* and *TopTwoSpokesAngle*.in all experiments. By graphing the logistic regression between these variables a lot of information can be used by the designer to improve the product. Figure 4.6 shows this logistic regression, the probability series represents the probability of a good outcome at every value of the independent variable (the design parameter). The affordance response series represents the evaluation from users. This means that if an angle of about 60 degrees is chosen for the steering wheel, then there is about a 65% chance that a user would rate it as a good design. A good design, as mentioned earlier, represents qualities of 1 to 3 based on the 7 point Likert scale used in the experiment.

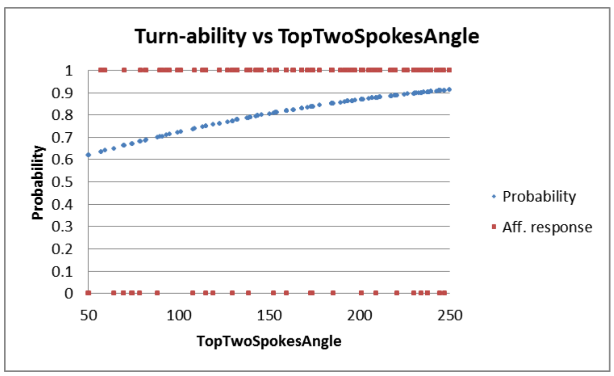
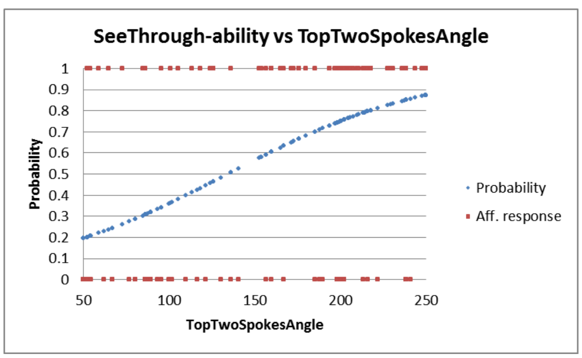


Figure 4.6 Turn-ability vs. TopTwoSpokesAngle Experiment 2

This can provide valuable information to a designer. Instead of trying to select one value for a design parameter the designer can focus on a range of values that would make the users have a positive perception of specific affordances in the product.

Besides helping designers target specific affordances with design parameter changes, these relationships give clues on how users make use of the product. The fact that *turn-ability* is related to the angle between the top two spokes may mean that users use the spokes to turn the steering wheel. This information could be used to identify new affordances that could improve the usability of the steering wheel. For example, designers could improve the grip on the spokes of the steering wheel to make it easier for users to turn the wheel or could make it such that users cannot use the spokes if they do not wish the drivers to use the spokes to turn the wheel.

Design parameters can be related to multiple affordances. Figure 4.7 shows the relationship between *SeeThrough-ability* and *TopTwoSpokesAngle* for experiment 1 (similar results obtained in experiments two and three). The results of this relationship makes sense as the larger this angle becomes, the better it is to see through the steering wheel, making it easier to see the gauges on the dashboard in the cabin. *HandRest-ability* is also related to *TopTwoSpokesAngle*, which means users think of resting their hands on the spokes rather than on the ring of the steering wheel.



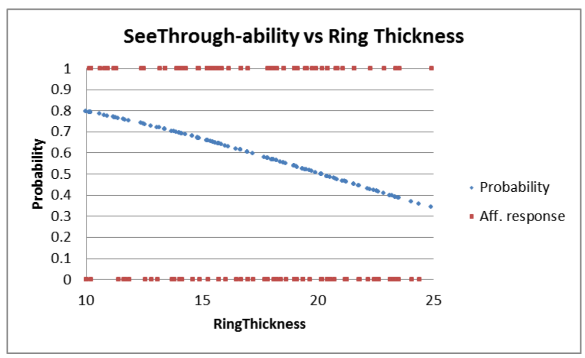


Figure 4.7 SeeThrough-ability/TopTwoSpokeAngle Exp. 1

Figure 4.8 SeeThrough-ability/RingThickness Exp. 2

The thickness of the ring (*RingThickness*) is related to *seeThrough-ability* and *protect-ability* (see Figure 4.8 for one of these relationships). Unlike the relationships shown earlier, this relationship is inversely proportional. This means that as the design parameter value increases, the probability of a good assessment decreases. In this case, the designer has to decide what to do as the trade-off between the two affordances would result in Pareto solutions, and the increase in one would lead to the decrease in the other (this is discussed more in depth in Chapter five).

Table 4.4 to Table 4.6 show the relationships obtained in each of the three experiments with real end users. Though there are common relationships in the experiments, there are relationships that only appear in one of the experiments. The relationships in each experiment directly depend on the perceptions of all the users in that experiment. Due to the fact that six users were used in each experiment, the results do not represent the perceptions of "all" end-users and therefore it is not expected that all experiments yield the same relationships. *The goal of these experiments is not to characterize the perceptions of all individuals*; for that to happen larger crowds need to be used in each experiment (which is possible with ABIGA); being careful in the selection of users so that they represent a large user base. However, the relationships found in all experiments do point to obvious perceptions from users, which might suggest that these relationships are expected to be seen in most experiments. The valuable takeaway is that the results prove that ABIGA can in fact extract the perceptions of groups of users through affordance/design parameter relationships.

Table 4.7 to Table 4.9 show the relationships obtained in each of the three experiments with a random input. Only one relationship was found in each experiment, the relationship is also different in each case. This proves that at least one of the relationships found in an experiment might happen out of chance. Given the number of possible relationships in an experiment this does not compromise the integrity of the set of relationships that are found with real users.

The next sections present how these relationships can be used to select design parameter values.

4.2.5.1 Choosing Design Parameter Values to Target Specific Affordances

The logistic regression is represented by Equation 4.1.

$$\ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x \qquad (4.1)$$

Where p is the estimated probability that a user would positively rate an affordance given a design parameter value of x. The intercept β_0 and the coefficient β_1 are given in the logistic regression results. With this equation and the results of the logistic regression for two variables the designer can determine the value of the design parameter for a desired probability of acceptance.

For the relationship shown in Figure 4.7, if the designer wants to know the angle between the top two spokes for which the probability of acceptance is 65%, equation 4.1 can be used to find it. This gives the designer a range of values for which the probability of acceptance is equal to or higher than 65%.

$$\ln\left(\frac{0.65}{1-0.65}\right) = -2.25213 + 0.0167285x$$

$$x = 171.633\ degrees$$

The result shows that if the designer chooses any value between 171.633 and 250 degrees, most users would rate the *turn-ability* of the steering wheel as good. This gives designers a lot of freedom when choosing values for different design parameters while making sure that their decisions will be perceived as good solutions by the users.

Note how the regression analysis can provide design parameter values that are outside of the tested range. It is not recommended to use such values as they might violate space constraints. Even if using these values does not violate any constraints, it is still recommended to run the experiment again with the modified design parameter ranges.

4.3 Use of ABIGA Results

As shown earlier, two types of results can be obtained from ABIGA:

1. Optimal solutions (defined by a set of design parameter values and fitness values)
2. Affordances and Design Parameters relationships

4.3.1 ABIGA Archive results

Optimal solutions are extracted from the IGA archive, which stores the best solutions found during the experiment. These solutions are fully defined by their design parameter values and the affordance quality values it received from the user evaluations. It is important to know that the results of ABIGA may not directly define the shape of the product, that is, designers may not directly use the concepts found in the archive. *The design of a steering wheel cannot solely depend on the input of users*. The results of ABIGA should be used as another evaluating criterion in the design process. It is also important to know when to use ABIGA, that is, know at which stage of the design process the tool should be used (discussed later in this section).

Looking at how the solutions evolved in an experiment can help designer identify trends in the shape of the solutions. The design parameters of the archive solutions can also give designers a starting point in determining the final design parameter values of their product.

Figure 4.9 and Figure 4.10 show a subset of solutions found in the first generation of the IGA and a subset of solutions found in the archive, respectively. From these results the designer could conclude that the angle between the top two spokes evolves to a wide angle. This could mean that wide angles offer better seeThrough-ability. Another trend that can be seen is the size of the hub. Early solutions have multiple sizes of hubs (as expected), but the archive shows that the size of the hub evolves to a small/medium hub. This may have other implications besides allowing better *seeThrough-ability*. It could be that a smaller hub offers more space for users to rest their hands on the spokes of the steering wheel when the vehicle is stopped, in other words, a smaller hub affords a better *handRest-ability*, which is also related to the angle of the top two spokes.

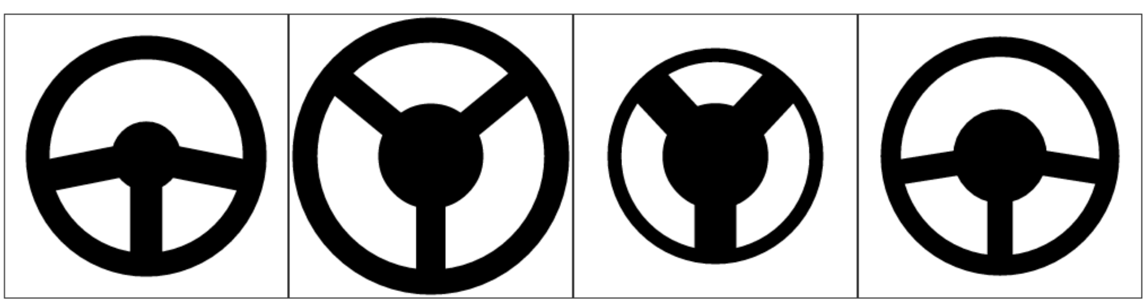


Figure 4.9 Generation 1 Solutions Experiment 1

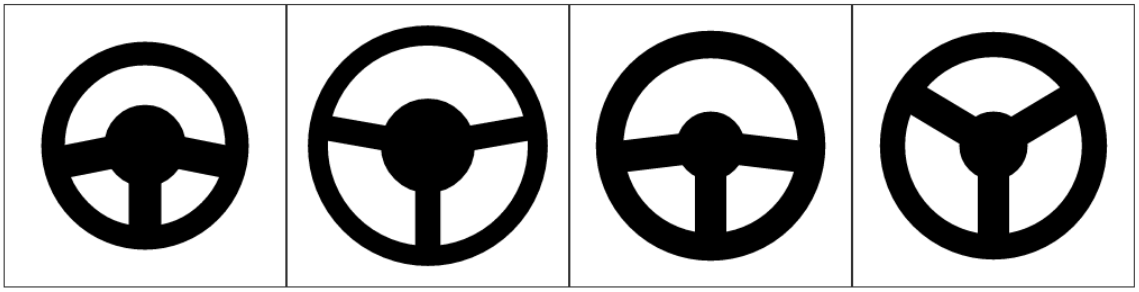


Figure 4.10 Archive Solutions Experiment 1

Figure 4.11 and Figure 4.12 show the same information as above from the results of the second experiment. When comparing the evolutions of the two experiments another trend can be spotted. The concepts tend to evolve toward medium sized steering wheels. Both examples have large steering wheels (large ring diameter) in the first generation, but the solutions in the archive show that the size of the steering wheel is reduced to a medium size. These of course are qualitative assessments of the evolution results obtained from ABIGA. This however does not mean that the information isn't helpful; it forces designers to look for trends that can only be obtained from user evaluations.

The archive results can give designers a starting point when choosing the design parameter values. These values can then be refined using the information obtained from the affordance/design parameter relationships (discussed later) and other evaluating criteria.

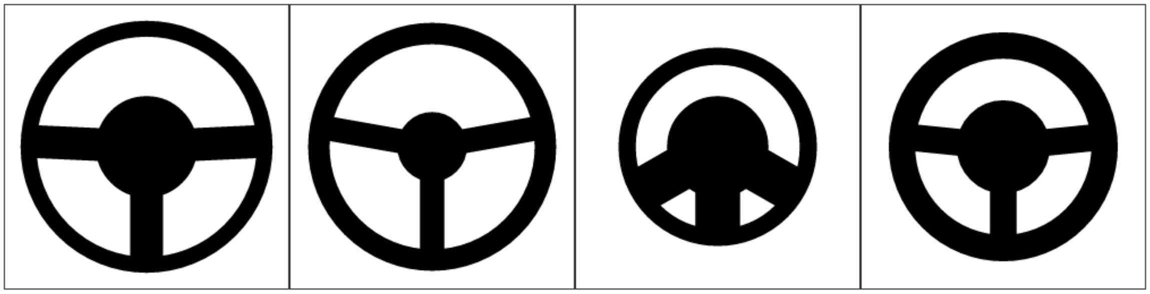


Figure 4.11 Generation 1 Solution Experiment 2

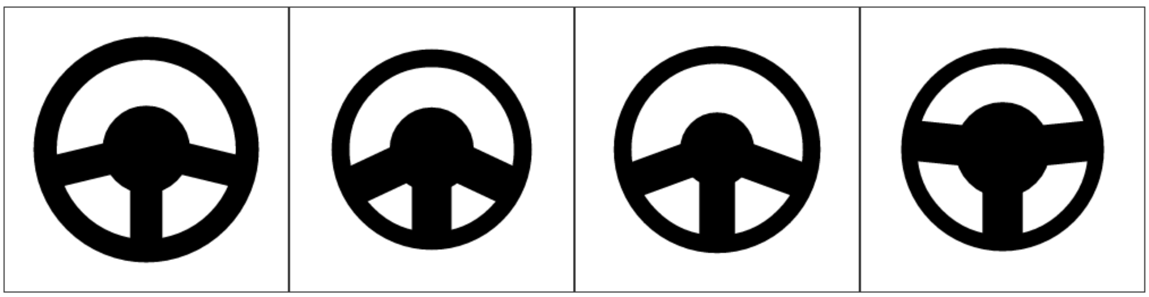


Figure 4.12 Archive Solutions Experiment 2

Note: Archive solutions do not all come from the last generation of the IGA. The archive's job is to save the best solutions seen throughout the entire optimization run. However, it is expected that most of the top solutions found in the archive do indeed come from the last generations of the IGA. This is because, as seen in the evolution results shown earlier, high score solutions are more likely to be in the generations that have the highest average fitness score.

4.3.2 ABIGA Affordance and Design Parameters Relationships Results

As introduced earlier, qualitative evaluations of the results can provide helpful information to the designer. However, if the results of other analyses (e.g., stress analysis, cost analysis) are in conflict with the design parameter values obtained from one of the archive solutions, then the qualitative assessment results will not help designers determine the set of design parameter that satisfy all evaluating criteria. This is where the relationships between affordances and design parameters can greatly improve designers' ability to modify design parameter values while still making sure that the changes made to the product's form still uses the information from the user input experiments.

So far it has been shown how affordance and design parameter relationships can be extracted from the user input experiments. These relationships (example shown in Figure 4.13) can be quantified using logistic regressions. An equation can be obtained for each affordance/design parameter relationship. The equation describes the likelihood that an end-user would rate the affordance as "good" for any given design parameter value.

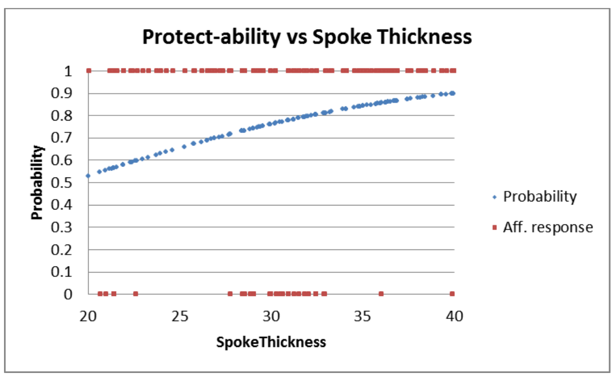


Figure 4.13 Protect-ability vs Spoke Thickness Experiment 2

Looking at the graphs generated by the logistic regression it is evident that designers don't necessarily have to choose a single value for their product's design parameters. This is because designers know that they cannot come up with a design that is preferred by all end-users. Instead, they would try to get positive affordance quality perceptions from most end-users, meaning they could set a minimum *User Acceptance Probability* (UAP) and find a range of design parameter values. This would give designers the possibility of changing the design parameter values within a range while making sure that any of those values would elicit positive user assessments of a particular affordance.

As an example, consider the relationship shown in Figure 4.13, this relationship suggests that the larger the thickness of the spokes, the higher the chance a user would rate the steering wheel's protect-ability as being 'good' (1 to 3 score). If the UAP is set at 70% (0.70), the corresponding design parameter value can be calculated using equation 4.1. Once this value is obtained, the range can be defined with either the minimum or maximum design parameter value.

From the logistic regression analysis:

$\beta_0 = -1.96306$

$\beta_1 = 0.10407$

The corresponding design parameter value can be calculated using equation 4.1:

$\ln\left(\frac{0.70}{1-0.70}\right) = -1.96306 + 0.10407x$

$x = 27.00450$

The sign of the relationship is defined by β_1. In this case the sign is positive. This means the probability of acceptance increases as the design parameter value increases. This is used to define the limits of the design parameter range. The range of design parameter values for this affordance becomes $[27.00, 40.00]$. This gives designers more freedom to change the design parameter values. These ranges can be determined for all design parameters that have at least one relationship with an affordance.

There are two possible scenarios when multiple relationships are present for a design parameter. The first scenario is when all of the relationships are of the same type (either all positive or all negative). It is possible to find a range that satisfies all of the affordances related to the same design parameter. The range in this case is defined by the intersection of all the ranges found in the relationships. This is described in equation 4.2.

$$Effective\ Range = Range_1 \cap Range_2 \cap \ldots \cap Range_n \qquad (4.2)$$

The second scenario is when different types of relationships exist for the same design parameter. Changes in a product's design parameters can have a positive impact on the perception of quality of one or more affordances, but it may negatively affect others. The affordances involved in these relationships are said to be *conflicting*. When conflicting affordances exist, an effective range cannot be determined, because the individual ranges found for different relationships might be at opposite ends of the full design parameter range.

To illustrate this scenario, Figure 4.13 shows one of three relationships related to the spoke thickness. The range of values that guarantee an acceptance probability of over 0.7 was $[27.00, 40.00]$. Figure 4.14 shows another relationship (for the same experiment, experiment 2) between *seeThrough-ability* and *Spoke thickness*. This relationship is negative, because the user acceptance probability decreases as the design parameter value increases. The range of values for a UAP of 70% in this relationship is $[20.00, 24.70]$. The intersection of these two ranges yields an empty set.

To solve this issue two ranges are shown to the designer, the effective range of all positive relationships and the effective range of all negative relationships. The designer needs to prioritize some affordances over the others. By selecting a value from one of the ranges, the designer is basically favoring the affordances related to the range used. This trade-off is common; an example of this is the size of cellphone screens. A large screen offers better *screen view-ability* than a small screen, but its *port-ability* is worse than a small screen cellphone.

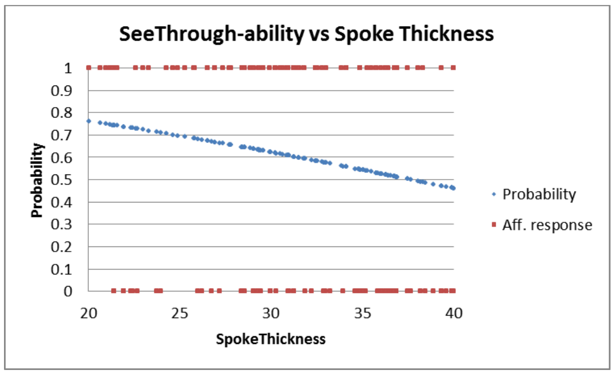


Figure 4.14 SeeThrough-ability vs Spoke Thickness Experiment 2

Though ABIGA does not automatically determine the effective ranges from the affordance/design parameter relationships, this process can be automated. However, because ABIGA was coded using the MVC (model-view-controller) architecture, adding this function to the application becomes easier.

4.3.3 Use of ABIGA in the Affordance Based Design Process

ABIGA evolves the shape of products towards optimized variants based on the input of end-users. This means that the product's architecture is well defined; the design parameters that define the shape are known by the designer. ABIGA is therefore meant to be used during the final stages of the ABD process.

Figure 4.15 shows the systematic approach to designing artifacts when using the concept of Affordances. Though product architectures are created in the early stages, these architectures do not have a well-defined set of design parameters, because the concepts are generally created as sketches. ABIGA should be used when designing AUAs at the final stage of the ABD process.

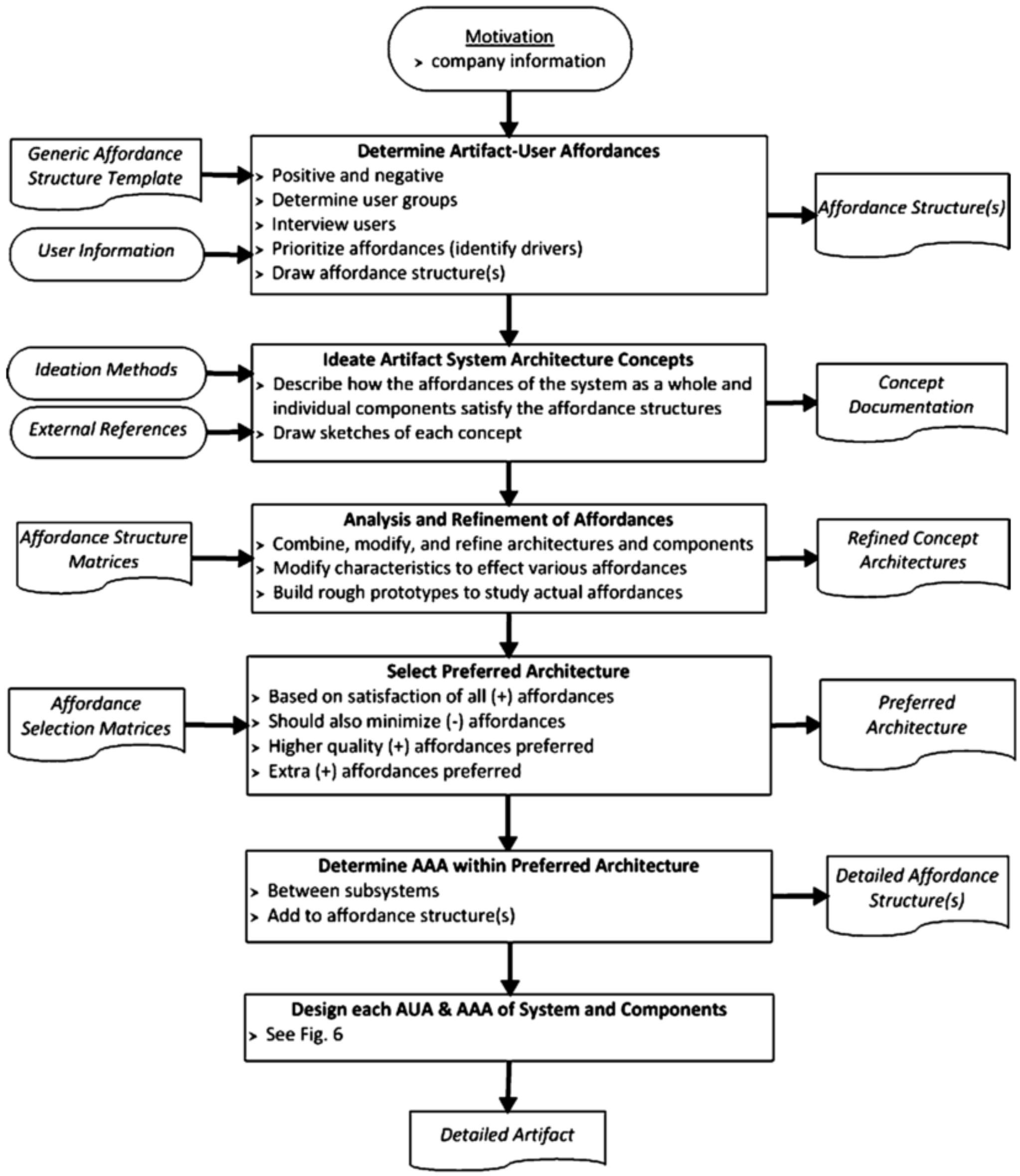


Figure 4.15 ABD Systematic Process (Maier & Fadel, 2009b) (repeated)

Figure 4.16 shows a systematic process on how to design the AUAs and AAAs of the artifact. It is at this stage where ABIGA can be used to its full potential. ABIGA can help designers with stages two, three and four of the AUA/AAA Design method.

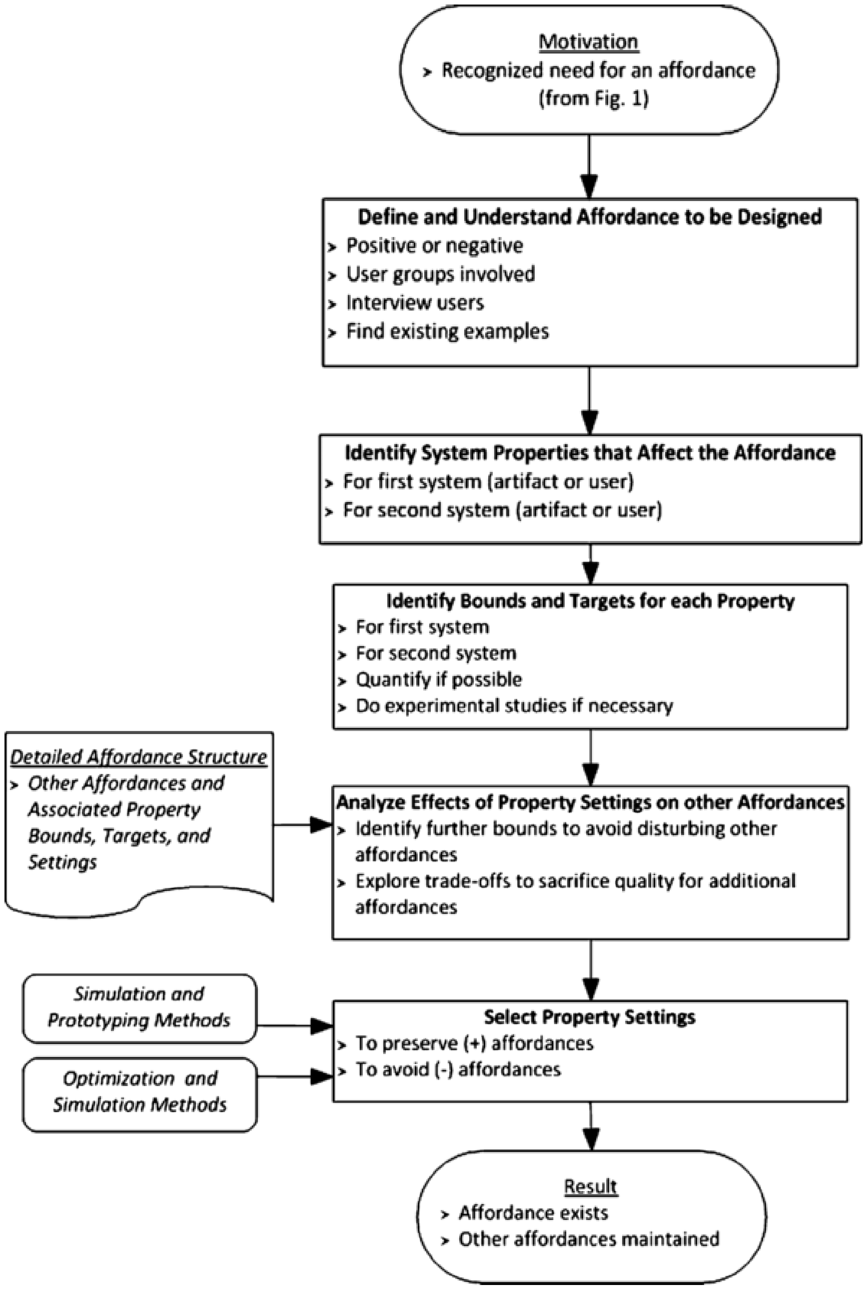


Figure 4.16 Affordance Design within ABD (Maier & Fadel, 2009b)

The first stage of this method deals with understanding the affordances of the product. ABIGA is not needed in this stage, because the designer can determine the type of affordance and the interacting entities without the input (in the form of affordance quality perceptions) of end-users. ABIGA should be used for the following three stages of this method:

- Identify System Properties that Affect Affordances
- Identify Bounds and Targets for each Property
- Analyze Effects of Property Settings on other Affordances

It has been previously shown how ABIGA results can relate the input from end-users with the architecture of the product. The user input is in the form of evaluations of quality perceptions, and the architecture is defined by the design variables of the artifact. Without ABIGA, the designer would most likely rely on all the potential physical interactions between the components of the product and the user. The relationships found by ABIGA directly link how the design parameters affect the affordance-quality perceptions from users. This means that if the designer were to make changes to the architecture of the product, he would know which affordances would get affected. This is important because it informs designers how potential changes might negatively affect certain affordances.

ABIGA results can also provide bounds for the design parameters. This was possible by doing logistic regression analysis on pairs of affordances and design parameters. Identifying these bounds not only helps knowing which values the design parameters could adopt based on user input but it also helps designers see how the different affordances are related to each other. Multiple affordances can be related to the same design parameter. When this is the case, the designer knows exactly how a change in that parameter affects the other affordances. It could be that a change in a design parameter improves an affordance while negatively affecting other affordances; this points to any possible trade-offs.

ABIGA offers a way to include the input from users when selecting design parameter values at the last stage of the ABD process. The process of designing single affordances becomes easier with the use of this tool because it offers a visual way to understand how the affordances are related to the architecture of the product that at the same time allows for a deeper understanding of how the different affordances of the product depend on each other.

4.3.4 Possible Use of ABIGA in the Early Stages of the Design Process

ABIGA has been shown to assist designers during the last stage of the ABD process; but it is possible to use it during the early stages of the ABD process after some modifications. User input can be used in the early stages to help designers in selecting product architectures. The limitation with the current build of the application comes from only being able to evolve the shape of the artifact and not its topology. In order for the application to be used early in the design process, designers need to explore various concepts, which may have different topologies.

The IGA/ABD integration allows for the optimization of a fixed number of objectives (affordances) and design variables (design parameters) once the optimization has begun. When the topology changes, the set of design parameters that is needed to describe each topology is different. For example, the steering wheel used at the beginning of the chapter has three spokes, but during the conceptual stage of the design process other options could have been explored. Adding just one spoke (that is, a steering wheel with four spokes) adds two new design variables, which are the angle between the bottom spokes and the number of spokes (three or four spokes). A way to handle this change in the number of variables during an optimization run is needed to allow ABIGA to work with different topologies in the same experiment.

If ABIGA is able to support varying topologies during an optimization run then it could be used to select a topology based on the input from end-users. Besides giving designer valuable information regarding the relationships between affordances and design parameters, the results would also alert designers how the topology is related to the different affordances. The results would show which topologies are more likely to elicit positive affordance quality perceptions from the users. The tool would therefore be used during the ***Select Preferred Architecture*** stage of the ABD process (see Figure 4.15). Of course this would not be the only method designers would use to select a concept, the use of the tool would be considered an extra selection criteria.

There would also be a difference in how the experiment's affordances are chosen. As a reminder, when used in the last stage of the ABD process, the designer needs to select a set of artifact-user affordances that will most likely have a big impact on the shape of the product. During the early stage of concept selection, the designer would need to choose a set of affordances that are common to all concept variants and that highlight comparisons between them. For example, consider three steering device concepts: a round steering wheel, a joystick and a squared steering wheel. A common affordance is that of *grip-ability*. However, *tilt-ability* is not, because it is only present in the joystick concept.

Considering that designers know the different concepts they want to explore, the design parameters needed to describe all the concepts can be combined into a single set (see equation 4.3). The idea is that the genetic algorithm will be initially set with all the variables needed to define all concepts. A variable that defines all architectures would also have to be added so that the GA selects the appropriate set of design parameters. The user interface would then generate the corresponding virtual prototype based on the variable that defines the architecture of the artifact

$$Design\ parameters\ set = DPS_1 \cup DPS_2 \cup \ldots \cup DPS_n \tag{4.3}$$

The IGA would be working with a large set of design parameters, which means that the number of design variables will be the same throughout the GA run, but the set of design variables used for every solution would differ. This is illustrated in the following example.

Consider the designer is evaluating three concepts, C1, C2 and C3. Each of which is defined by a design parameter set (DPS). The affordance set is the same for all concepts.

$C1$:	$C2$:	$C3$:
$DPS1_1$	$DPS2_1$	$DPS3_1$
$DPS1_2$	$DPS2_2 = DPS1_2$	$DPS3_2 = DPS1_2$
$DPS1_3$	$DPS2_3$	$DPS3_3$
$DPS1_4$		$DPS3_4$
		$DPS3_5 = DPS1_4$

The overall set of design parameter when combined becomes:

$$DPS = \begin{Bmatrix} DPS1_1 \\ DPS1_2 \\ DPS1_3 \\ DPS1_4 \\ DPS2_1 \\ DPS2_3 \\ DPS3_1 \\ DPS3_3 \\ DPS3_4 \\ C \end{Bmatrix}$$; Where "C" defines the type of concept architecture.

ABIGA would then be used to help designers evaluate all the concepts. The ABIGA experiment would be set using the overall set of design parameters and a common set of affordances. ABIGA would check the "C" variable of the solution and generate a drawing with the corresponding set of design parameters.

There are more changes that need to be made to have ABIGA handle multiple artifact architectures in one experiment. As described in Chapter 3, the virtual model of the product is generated with JavaScript code. If virtual models are generated in the same way, there would need to be a JavaScript code that can draw all the different product architectures, organized in drawing functions, one per product architecture. The variable that defines the product architecture would be used to specify which drawing function to use.

4.4 Summary

This chapter provided details about the implementation of ABIGA. As a proof of concept, a steering wheel was redesigned using the application. Three experiments were done using real people as the affordance quality evaluators and three experiments were done using a random input. The results showed that ABIGA can indeed evolve designs toward better perceived variants when real users act as the evaluator. The steering wheel did not evolve toward better solutions when the input was randomized. This proves that the IGA cannot evolve products out of chance.

It's important to understand that the evolution in ABIGA only affects the form of products. The results are applicable to the geometrical characteristics of products. How the evaluations from users affect the internal aspects of products is not studied.

Besides giving designers optimized solutions, another important result of ABIGA is the ability to identify relationships between the affordances of a product and its design parameters. The relationships provide insight on how specific design parameter values affect the user perception of affordance qualities.

ABIGA is not intended to be the only tool needed to redesign a product. The solutions found in the archive can be used as the initial set of design values which can later be changed according to other evaluating criteria. It was found that with the results designer can identify ranges of values for each design parameter that has at least one relationship with one of the affordances. These design parameter ranges are determined after specifying a user acceptance probability, meaning any value within the range maps to an acceptance probability that is higher than the one specified. This gives designers the freedom to change the values of the product's design parameters to values that have a high chance of being perceived to be good by users.

It was demonstrated that ABIGA should be implemented in the last stage of the ABD process. This is because the architecture of the product has been fully defined, meaning the designer can select a set of design parameters that define the shape of the product. It was also suggested what the changes ABIGA needs to support its use during the early stages of the design process. The changes include the ability to handle multiple product architectures within the same experiment. This can be solved by identifying the set of design parameters that define all product architectures and by adding a variable that identifies each configuration. If implemented, these changes could help designers during the concept selection of the design process with the perspective from end-users while still providing valuable information regarding affordance and design parameters mappings.

Chapter 5

5 A Retrospective Analysis of the Design of ABIGA experiments[4]

Chapter 4 dealt with the redesign of a steering wheel and how the results from ABIGA can aid designers during the last stage of the ABD process. This chapter will examine how important the setup of the experiment is to achieving good product evolutions. The setup of the experiment includes the selection of the set of affordances and design parameters of the product as well as the way the experiment should be distributed to users.

Unsatisfactory results obtained in a separate product evolution experiment led to the analysis of how the choice of product affordances and design parameters could affect the evolution of the product in the experiment. Other factors that were studied were the

---

[4] In collaboration with Mo Chen,
Department of Mechanical Engineering, Southeast University, Nanjing, 210000, China

quality product representation and the user-access strategy implemented. There of course can be many more factors that could potentially affect product evolution when using AB-IGA, but only those that could be supported with evidence were studied.

5.1 Redesign of a Compact Digital Camera

Besides the steering wheel proof of concept experiment, a compact digital camera redesign experiment was also conducted. The camera was modeled with six design parameters (see Table 5.1). The shape of the camera is shown in Figure 5.1. Users were shown three two-dimensional views of camera concepts: top view, front view and back view. Five users were asked to evaluate camera variants. The affordances of the camera are shown in Table 5.2 along with their descriptions.

Table 5.1 Camera Design Parameters

Design parameter	**Minimum value**	**Maximum value**
Width	325 pixels (86 mm)	430 pixels (114 mm)
Height	150 pixels (40 mm)	300 pixels (79 mm)
Depth	50 pixels (13 mm)	160 pixels (42 mm)
Fillet Radius	0.10 (1.3 mm)	0.40 (17 mm)
Screen Size	0.50 (20 mm)	0.70 (55 mm)
Lens Size	0.50 (20 mm)	0.95 (75 mm)

The last three design parameters in the table do not show units because they are scale factors. These design variables depend on the width, height or depth of the camera.

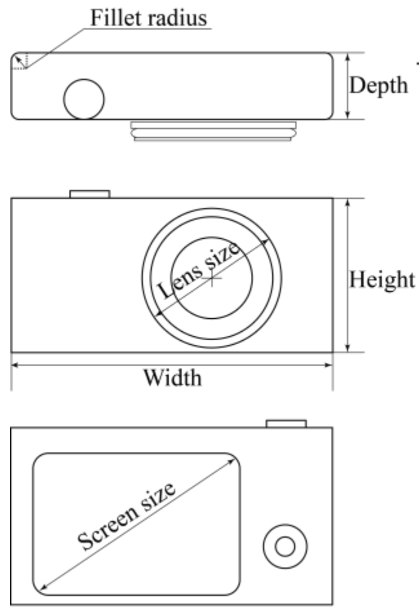


Figure 5.1 Camera 2D drawing

Notice how not all affordances in Table 5.2 are AUAs, one of the affordances, *exposure-ability*, is actually an AEA. The reason this affordance was chosen is because its quality is related to a design parameter (lens size) that the users can perceive when evaluating the product. The designer here assumes the users know of the relationship between lens size and exposure-ability.

Table 5.2 Camera Affordances and Descriptions

Affordance	**Description**
Port-ability	Interaction between the user and the camera when users carry it by hand or store it in a bag/pocket.
Hold-ability	Holding interaction between one hand or two hands of the user and the camera.
Stability	The likeliness that the camera can be knocked over from an upright position.
Exposure-ability	Exposure is the unit of measurement for the total amount of light permitted to reach the electronic sensor during the process of taking a photograph.
Screen view-ability	Visual interaction between the user and the camera that allows the user to conveniently see the photos to be taken on the screen.

The first camera experiment was conducted with a set of five users. The way these users accessed the application differed from the way users accessed the application when evaluating the first two steering wheel experiments introduced in Chapter 4. The third steering wheel experiment and the camera experiment were done simultaneously. Unlike the first two steering wheel experiments, users were not simultaneously asked to evaluate design concepts. This means that the likelihood of having multiple users assess affordances concurrently was significantly reduced. The possible effects of having users access the application as previously described will be discussed later.

5.1.1 Camera Redesign Results

The camera experiment was run for twenty generations. The evolution results are shown in Figure 5.2. Even though the camera's solutions in the last generation are similar to the solutions in the first generation, there is an improvement tendency across all generations. However, the camera experiment did not achieve obvious improvements as the ones seen in the steering wheel experiment (see Figure 5.3).

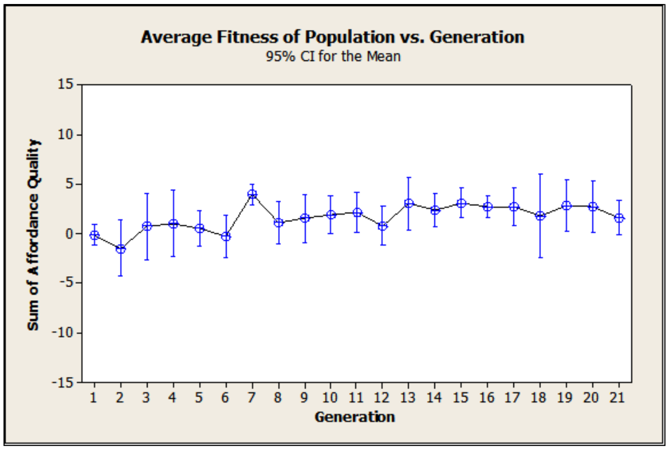


Figure 5.2 Camera Evolution Experiment 1

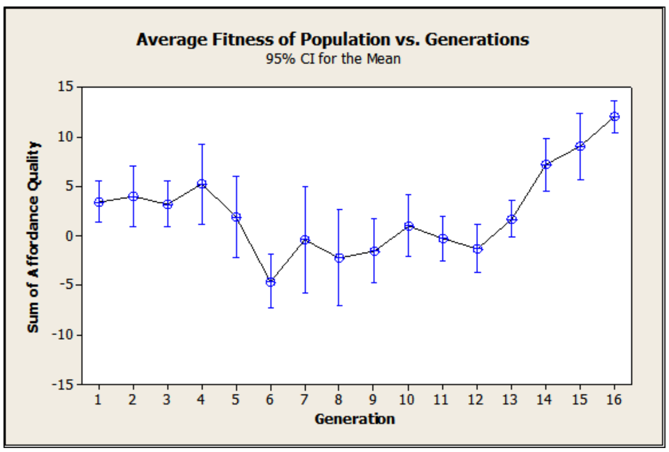


Figure 5.3 Steering Wheel Evolution Experiment 1(repeated)

Based on these results it can be assumed that the steering wheel results improve at a higher rate than the camera results. There might be multiple reasons as to why this happened. Based on the analysis of the evolution of the camera and steering wheel experiments, two possible reasons are hypothesized as to why the camera experiment results did not evolve at the same rate as the steering wheel results did.

5.1.2 Hypothesis one: Use of Conflicting Affordances

Changes in a product's design parameters can have a positive impact on the perception of quality of one or more affordances, but it may negatively affect one or more other affordances. When this happens, it is said that the affected affordances are conflicting. *The selection of conflicting affordances hinders the evolution of products in a GA/affordance framework*. This is a common issue in multi-objective optimization, where the goal of the optimization algorithm is to generate Pareto optimal solutions. Pareto solutions would emphasize finding a trade-off between the affordances. If the designer does not want a trade-off, it may be beneficial to identify ways to avoid conflicting affordances by modifying the design.

5.1.3 Hypothesis two: Use of a Weak Scale Reference

The quality of the representation of the product plays a very significant role in the ability of users to perceive the affordances they are rating. Seeing an object on a screen does not necessarily convey to the user how big or small it is in real life. A representation that uses multiple views as understood by engineers might not be the right way to solicit input from users who do not have the experience to form a 3D image in their mind based on multiple views.

5.2 Validation of Hypotheses

5.2.1 Existence of Conflicting Affordances

To validate hypothesis one, it is necessary to show that the affordances used in the camera experiment are conflicting and that improvements can be attained once the conflict in affordances are minimized. The same data analysis methods used in Chapter 4 can be implemented. From Chapter 4, it can be concluded that relationships between affordance evaluations and the design parameters of products can be obtained from user evaluations. These relationships can help designers determine if the experiment had conflicting affordances by examining the sign (positive or negative) of the relationships associated to the same design parameter. If two or more affordances that are related to the same design parameter have different signs, then it means these affordances are conflicting. This is because a change in the design parameter would show how different affordances diverge in quality value.

Table 5.3 Binary Logistic Regression P-value Results Camera Experiment 1

	Width	Height	Depth	FilletRadius	ScreenSize	LensSize
Hold-ability	0.000	0.350	0.001	0.619	0.854	0.436
Stability	0.296	0.160	0.000	0.243	0.000	0.313
Exposure-ability	0.014	0.000	0.802	0.835	0.002	0.000
Portability	0.001	0.009	0.000	0.011	0.106	0.390
ScreenView-ability	0.033	0.198	0.384	0.198	0.000	0.002

Table 5.3 shows the results obtained from binary logistic regression analyses to check for relationships between the design parameters and the affordances of the camera. Even though there are many affordance/design- parameter pairs that are related, this does

not mean they are all important to the designer. Some pairs will have a relationship because of the way the design parameters were selected. For example, *exposure-ability* is related to the lens size of the camera. This makes sense as the size of the lens determines how much light is captured and focused on the sensor of the camera. However, *exposure-ability* shows to be also related to the height and the width of the camera, which does not seem to make sense until it is observed that the lens size may be relative to the height of the camera. This in fact shows how powerful the application can be at identifying relationships the designer might not have expected. For the same reason, other indirect relationships can also be neglected: stability/ screen size and screen view-ability/ width.

Some of the most important relationships are explored next and it is shown how this information can prove the existence of conflicting affordances.

5.2.2 Stability, Hold-ability and Portability versus Depth

Graphing the logistic regression between two variables shows the probability that users would perceive a positive affordance quality for particular values of a design parameter. Figure 5.4 shows the graph of the logistic regression between Portability and Depth. The graph shows two sets of data. The affordance response (red dots) is the categorized user response as explained in Chapter 4; a “one” means a user deemed the portability as good while a “zero” means a user perceived it as not good. The probability data set (blue dots) represents the probability that a user would rate a product's affordance as "good".

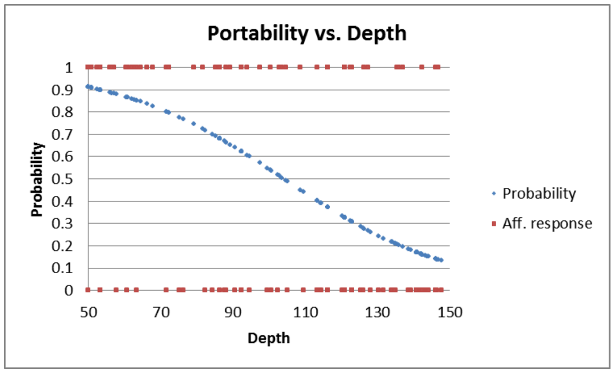


Figure 5.4 Portability vs Depth Camera Experiment 1

The graph suggests that large values of depth for the camera would most likely make the portability quality of the camera be perceived as bad. This is because the bigger the depth of the camera the harder it becomes to carry it around. For example, if a value of around 130 is chosen for depth, there is approximately a 20% probability that users would think this is a good quality camera regarding portability, meaning the vast majority of users would deem its portability as bad. The same tendency can be seen in Figure 5.5 when comparing the hold-ability of the camera with its depth. This makes sense as the larger the depth of the camera is, the harder it becomes to hold. However, the effect of depth on hold-ability is lesser than the one on portability.

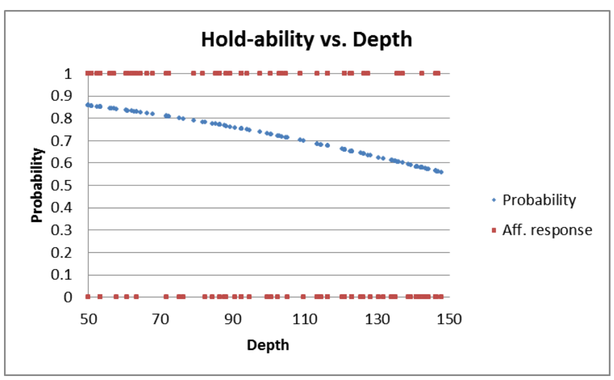


Figure 5.5 Hold-ability vs Depth Camera Experiment 1

When analyzing the relationship between stability and depth (see Figure 5.6), it can be seen that as the depth of the camera increases, the better its stability is perceived by the users. This also makes sense, because the larger the depth the better is its ability to stand on its own.

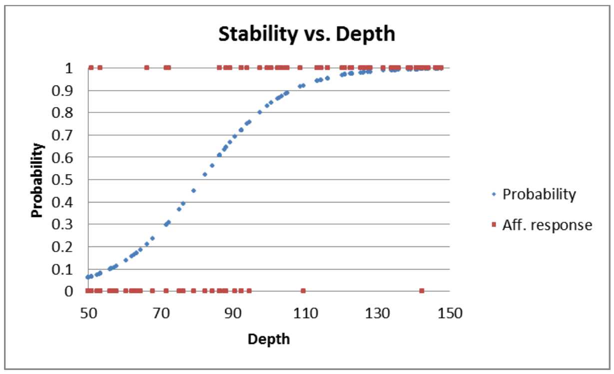


Figure 5.6 Stability vs Depth Camera Experiment 1

This opposite effect of depth on multiple affordances shows that the multi-objective aspect of the GA is functioning well. Solutions may eventually become Pareto optimal and there will be a trade-off between affordances. When the GA changes a parameter that improves one affordance, say stability, this negatively affects other affordances such as hold-ability and portability. Thus stability is in conflict with hold-ability and portability. Similarly, exposure-ability and portability are shown to be conflicting. Overall, four out of five objectives are in conflict with at least another objective. This would explain why the GA did not show consistent improvement in the affordances when evolving the design concepts across the different experiment generations in the case of the camera.

5.2.3 Quality of the Virtual Product Representation

The more senses users get to use in a virtual environment, the better they will be able to assess the quality of product affordances. The ideal virtual environment would be one where a user could see and touch the object. This may be possible with current technology (VR headsets, haptic mice, etc.), albeit expensive technology, but our first proof of concept focuses on visual representations of products which users can interact with using their computers and phones.

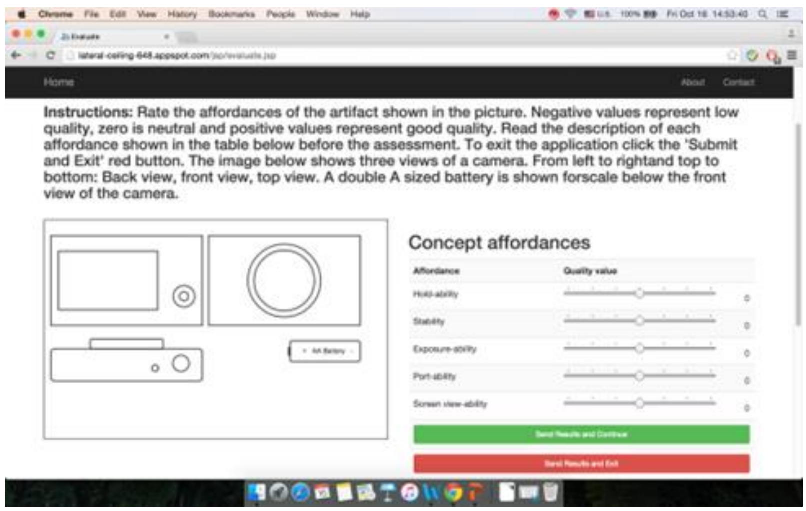


Figure 5.7 Camera User Interface for User Evaluation

Reflecting on the steering wheel and camera experiments, placing the product in the environment where it is normally used worked for the steering wheel problem. Figure 3.10 shows that the steering wheel is placed in the dashboard of a car where it becomes easy to understand how big it is. For the camera experiment (see Figure 5.7) the size reference used was a double-A battery and the camera was represented using three views. The drawing was not placed in an environment similar to the one of the steering wheel. This, we presume, makes it harder for users to perceive the size of the object and consequently affects the evolution of the product since there is no clear reference for the various users to be consistent.

Based on these observations, some guidelines are proposed next that aim to minimize the effects of conflicting affordances and improve the quality of the product representation. The modified camera experiment results will be shown afterwards.

5.3 Guidelines for Preparing ABD User Experiments

5.3.1 Guideline 1: Predicting Conflicts between Affordances though Design Parameter Relationships

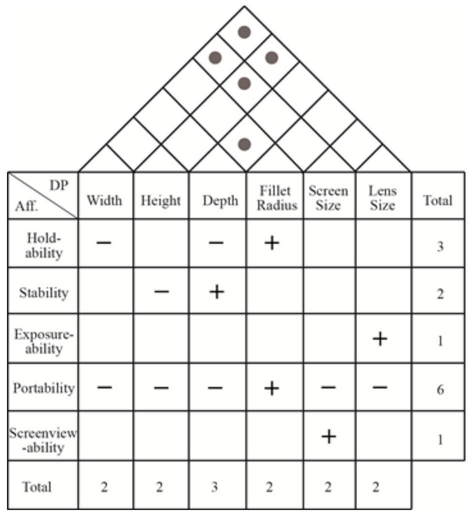
As the camera results suggest, designers should attempt to avoid having many conflicting affordances in their experiments. This seems to be in conflict with the idea of Pareto optimization; however, such a situation may be the impetus to help the designer come up with alternative solutions as explained in next.

It is a challenge for the designer to design products that have no negative affordances. Is there however a way to change the design to eliminate the negative affordances? One way to avoid having conflicting affordances is to predict their existence when the sets of design parameters and affordances have been determined so that the designer can select a set of affordances with minimized conflicting affordances.

The detection of conflicting affordances prior to running the experiment with real users will depend on the designers' experience. However, there is an Affordance Based

Design tool that can be modified and implemented to help designers in the process. The Affordance Structure Matrix (ASM) is such a tool designed to help in the variant design process as a means to identify the components of the artifact that need improvement based on their relationship with the affordances of the product (Maier & Fadel, 2007). Later, they extended the original formulation of the ASM by considering whether each component has a helpful (+), harmful (-) or no () relationship to each affordance. In order to indicate how much room there is to improve a component, they further introduce the relative percentage of helpful to harmful relationships. For example, a negative percentage value for a component implies the component might need to be optimized or redesigned in the future.

Based on the extended ASM (Maier, Sandel, & Fadel, 2008), a modified ASM can help identify relationships between the affordances and design parameters of the product. In order to predict potentially conflicting affordances, the modified ASM is created and populated *before the experiment is run*. Therefore, the modified ASM relies on the designer's experience and Figure 5.8 illustrates such an ASM for the Camera example. The design parameters are placed in the columns of the matrix and the affordances are placed in the rows. All parameters are assumed to increase in the example shown. If the affordance is believed to be rated as good then a "+" sign is placed in the intersecting cell. A "-" sign is used when the predicted assessment is negative. The roof of the modified ASM describes the relationship between design parameters. This part is vital to explain the existence of co-dependent relationships such as width/ screen size and height/ screen size. For example, the screen view-ability is related to width in the logistic regression, which seems irrational; in fact, it makes sense because of how the design parameters depend on each other as shown in the roof of the modified ASM.



DP / Aff.	Width	Height	Depth	Fillet Radius	Screen Size	Lens Size	Total
Hold-ability	−		−	+			3
Stability		−	+				2
Exposure-ability						+	1
Portability	−	−	−	+	−	−	6
Screenview-ability					+		1
Total	2	2	3	2	2	2	

Figure 5.8 Modified ASM to Predict Affordance Conflicts

The relationships between affordances are not listed on the left side of the matrix, which is slightly different from the ASM built by Maier. In the modified ASM, the relationships between affordances are identified through a design parameter.

After all affordances of interest have been evaluated, the designer needs to check for columns that have both positive and negative signs. The existence of opposite signs under a design parameter point to potentially conflicting affordances. Once these are identified, the designer can either:

- Accept the trade-off
- Favor one affordance over the other by possibly removing or replacing some affordances to minimize the number of conflicting affordances
- Modify the design to avoid the conflict

For example, from Figure 5.8, there are affordance conflicts related to three design parameters: depth, screen size and lens size. The row sum denotes the number of parameters that affect an affordance. All the parameters affect portability in our example, whereas single parameters affect screen-view-ability or exposure-ability. Designers can decide to substitute affordances based on this row sum. The results from the figure suggest that, based on the columns showing potentially conflicting affordances, one could consider not asking users to perceive the affordances of stability, exposure-ability or screen view-ability. However, dealing with either exposure-ability or screen view-ability only eliminates one conflict each. On the other hand, not considering the stability affordance would eliminate two conflicts. Note that the designer may still want to consider these affordances. A possible solution then would be to give more weight to certain affordances. This would explain why cellular phones have gotten bigger; focusing on the readability of the screen and consequently its size, putting less emphasis on the effects it might have on other affordances such as portability.

It's important to understand that using this tool will not necessarily help eliminate all conflicts between the affordances of the product. After the experiment is run there is still a chance that some unseen conflicts arise. However, the tool would still help designers identify obvious affordance conflicts.

5.3.1.1 Mitigating Affordance Conflicts by Changing Design Features

In the previous step, the stability affordance created a conflict. The designer could decide to add a rotating-base (see Figure 5.9) to make the stability of the camera independent of its depth. Thus, adding a design feature to the product could be used to eliminate relationships between affordances and specific design parameters, relationships that cause conflicts between affordances. This approach is feasible in our case because the users perceived the camera's stability to be under static conditions rather than in a dynamic state. In other words, the camera's stability was perceived to be its ability to remain stable on a surface such as a table while taking pictures. No change in parameters would have enabled them to perceive dynamic stability.

A similar method can be used for other affordances if the parameters that affect them are conflicting. For example, a strap hole can be added to the body of camera, which would make its portability independent of the corresponding design parameters under certain use conditions.

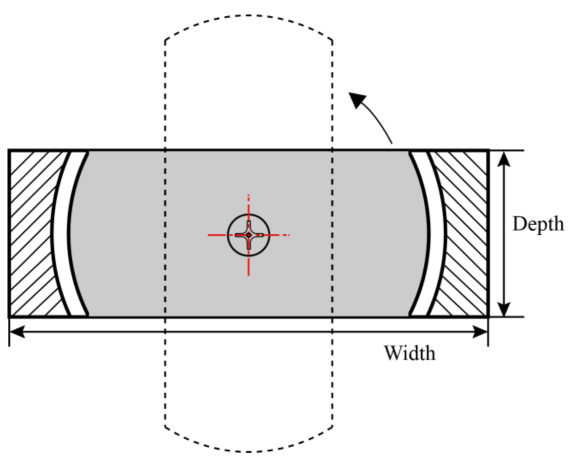


Figure 5.9 Bottom View of Camera with Added Stability Feature

5.3.2 Guideline 2: Use of an Effective Size Reference with Virtual Product Representations

The virtual-environment perception is strongly determined by the reference or background (context) shown along with the virtual object. An appropriate context can aid during information processing (Wolff & Wogalter, 1998). Therefore, one should use an appropriate background showing the environment where the target product will be placed. How can one identify if a specific context is good or not? Our examples highlight two aspects: The steering wheel is a component in a larger product, whereas the camera is an independent product. The steering wheel can be perceived clearly when displayed with a dashboard as background. The camera needs some other common object to aid the users in assessing affordances that depend on size. In our example, an AA battery was provided next to the 3 views of the camera, but after processing the results, we believe it was not obvious enough to help the users (maybe a real photograph would have helped). Also, for a product like the camera, the reference image should not misguide the user. For example, if a hand is suggested to be used as size reference (see Figure 5.10) it may imply to users that the perceived hold-ability of the camera is only dependent on its height.

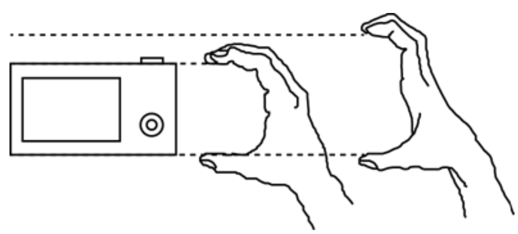


Figure 5.10 Different Hand Images as Size References

The camera representation used was a 2-dimensional orthographic figure. Reid et al. (Reid, MacDonald, & Du, 2012) confirmed that experimental results are not affected by the product form presentation (simplified forms vs. detailed realistic models) if judgments belong to inferences evaluation. They do for an objective evaluation. Since our experiments include both aspects, it would behoove the designer to improve the ability of the users to perceive affordances by adding some other design elements, such as shading, material and color. Furthermore, a 3D isometric representation may be more meaningful to inexperienced CAD users than orthographic representations.

5.3.3 Application of Guidelines to the Camera Experiment

Having identified two guidelines to set up ABIGA experiments, this section shows the results of implementing these guidelines on the evolution of the camera experiment. ABIGA aims to make use of visual representations of products which users can interact with using their computers or phones. Thus, simplified forms (2-D orthographic drawings) are still used in the application since they are easy to load. The evaluation web page still displays the 2D representation of the camera but with a modified reference that is a hand (see Figure 5.11).

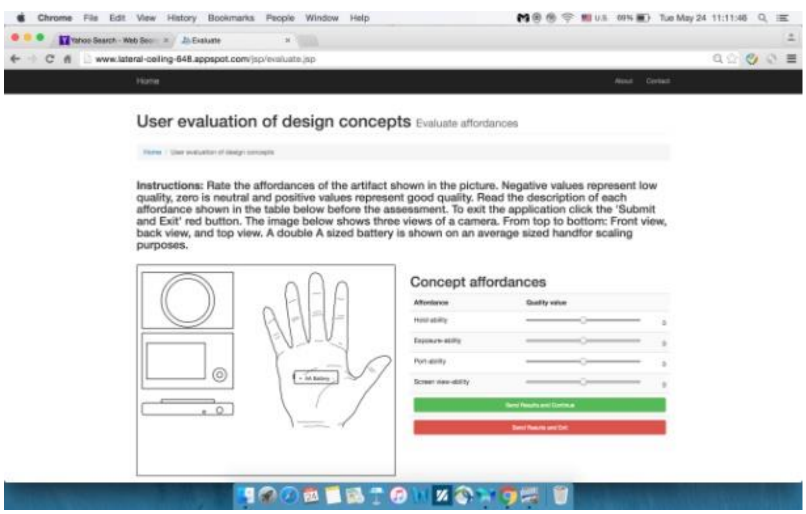


Figure 5.11 Modified Camera User Interface for Evaluation

Users' hand(s) necessarily interact with a camera. We infer that users will have a better perception of the dimensions of the camera when comparing it with a reference hand. In this experiment, the fiftieth percentile hand representation is used (for male and female combined). The average breadth and length of this hand are 88.4mm and 174.9mm, respectively.

One affordance, *stability*, was eliminated. This does not get rid of all the conflicts in the experiment, but it is a way to minimize them. The fitness of the entire population was also used to check the progress of the solutions across different generations. From Figure 5.12, results show that camera evolution has improved significantly when compared with the first experiment (see Figure 5.2), which supports that the proposed guidelines can improve the evolution. But the evolution is still not obvious when compared with the steering wheel results. Here, it cannot be ignored that the two cases represent different kinds of products. The steering wheel belongs to mechanical products, which provide many clues about its use. Users can directly and easily process information on interacting by relevant constraint factors. In other words, the structure of mechanical products is simple, even single, which results in the affordances being clear. However, the compact digital camera is one of consumer electronics; its appearance just plays a role of "package" or "wrapping". The relationship between "form or structure" and "affordance" can be diverse. The different ways in which users interact with a camera, to a certain degree, might have an effect on the overall user perceptions of affordance qualities. Nonetheless, this result is enough to verify that the proposed guidelines are effective.

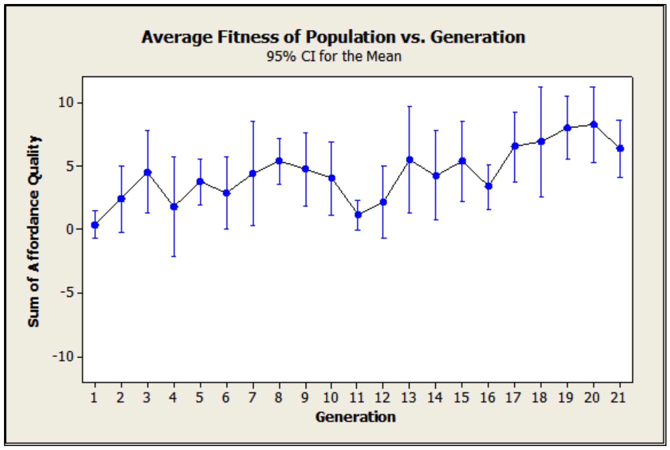


Figure 5.12 Modified Camera Experiment Evolution

5.4 Guidelines for Checking ABIGA Results

After running the experiments, interesting observations were made. These may be used to further define guidelines for the ABD experiment.

5.4.1 Independence Axiom from Axiomatic Design

In engineering design, the concept of functional requirements (FRs) is defined as "what we want to achieve" (Suh, 1998). These functional requirements are related to a set of design parameters (DPs). The Axiomatic design matrix (DM) links the functional requirements to the design parameters (see equation 5.1). The first axiom, the Independence Axiom, states that the independence of functional requirements should be targeted when designing a product (Suh, 1998).

$$\{FR\} = [DM]\{DP\} \tag{5.1}$$

5.4.1.1 Ideal Design

The basic requirement for the ideal design is that the number of FRs should be equal to the number of DPs. The design can be defined as coupled if the number of DPs is smaller than the number of FRs. When there are more DPs than there are FRs, the design is called a redundant design (see equation 5.2) (Park, 2007; Suh, 1998).

$$\begin{bmatrix} FR_1 \\ FR_2 \end{bmatrix} = \begin{bmatrix} X & X & 0 \\ 0 & X & X \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \end{bmatrix} \tag{5.2}$$

There are two kinds of matrices that describe the independence of function requirements. The first one is when the DM is a diagonal matrix; this means the design is uncoupled (equation 5.3). The other case is when the DM is triangular (equation 5.4). The design is said to be decoupled in this case (Suh, 1998).

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \end{bmatrix} = \begin{bmatrix} X & 0 & 0 \\ 0 & X & 0 \\ 0 & 0 & X \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \end{bmatrix} \tag{5.3}$$

$$\begin{bmatrix} FR_1 \\ FR_2 \\ FR_3 \end{bmatrix} = \begin{bmatrix} X & 0 & 0 \\ X & X & 0 \\ X & X & X \end{bmatrix} \begin{bmatrix} DP_1 \\ DP_2 \\ DP_3 \end{bmatrix} \tag{5.4}$$

5.4.2 Applying the Independence Axiom to ABIGA Experiments

Axiomatic design relates functional requirements to the design parameters of the artifact. Affordances can be related to the design parameters of the product in the same way due to their dependence on form (an affordance property). The relationship can be expressed by equation 5.5. The matrix A maps the Affordance (Aff) and design parameters (DP) relationships:

$$\{\text{Aff}\} = [\text{A}]\{\text{DP}\} \tag{5.5}$$

5.4.3 Comparing the Aff/DP Relationships in the Two Experiments

Logistic regression was used to identify the relationship between the affordances and design parameters. The matrix A is therefore based on the relationships found through logistic regression. The relationships for the steering wheel experiment were shown in Chapter 4 and the relationships for the camera experiment (not the modified experiment) were shown earlier in the chapter. Here, "1" represents a significant relationship, which indicates there is a dependency between an affordance and a design parameter. Zero indicates that the two are not related. From the two following equations 5.6 and 5.7, we observe that the design matrix of the steering wheel example is a triangular matrix, however, the design matrix of the camera example is not only a full matrix, but also the number of DPs is higher than the number of Affordances. Therefore, we can determine that the camera experiment is coupled.

$$\begin{bmatrix} \text{Hold} - \text{abillity} \\ \text{Stability} \\ \text{Exposure} - \text{ability} \\ \text{Portablity} \\ \text{Screen view} - \text{ability} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \text{Width} \\ \text{Height} \\ \text{Depth} \\ \text{Fillet radius} \\ \text{Screen size} \\ \text{Lens size} \end{bmatrix} \tag{5.6}$$

$$\begin{bmatrix} \text{Grib} - \text{ability} \\ \text{Turn} - \text{ability} \\ \text{Handrest} - \text{ability} \\ \text{Protect} - \text{ability} \\ \text{Seethrough} - \text{ability} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \text{Toptwo spokes angles} \\ \text{Ring thickness} \\ \text{Hub diameter} \\ \text{Spoke thickness} \\ \text{Spoke length} \end{bmatrix} \quad (5.7)$$

5.4.4 Guidelines to Decoupling Design Problems

As mentioned earlier, based on Suh's axioms, the ideal design problem is the one which has the same amount of affordances as design parameters. However, achieving a diagonal matrix in a design problem is often a challenge. The results shown in our two examples suggest that having a decoupled system can have a positive impact on the evolution of the product. Obviously, the two experiments are not sufficient to draw definite recommendations, but the parallel with axiomatic design lends credibility to our hypothesis: The designer should aim to have a decoupled system such as the one described in equation 5.4.

In an ideal design, the precondition is to make the number of Affordances (Affs) and the number of DPs the same. Any extra DPs should be deleted or several DPs should be merged. Another suggestion is to delete the design parameters that have the least amount of relationships with the affordances. They may not even be relevant to the designer when focusing on specific affordances.

If there is an insufficient number of DPs, the design is coupled, leading to some affordances that cannot be satisfied. Therefore, in order to keep the number of DPs equal to the number of Affs, some Affs that do not have much influence should to be deleted. The modified ASM shown earlier can help determine which affordances have the least influence. Another strategy is for some DPs to be added to make the number of Affs equal to DPs.

5.5 Summary

Conflicting evolutions between two design examples using ABIGA enabled the critical study of why this happened. It was found that most of the affordances that were chosen for the camera experiment were in conflict with at least another affordance by checking relationships between design parameters and affordances. This led to the hypothesis that the existence of conflicting affordances would result in a slower evolution when compared with one that did not have such conflicts. Having conducted multiple runs to support the hypotheses, it was observed that virtual presentation does influence the users' judgments. Furthermore, studying the relationships between affordances and design parameters, Axiomatic design's independence axiom was used to further explain such a behavior. From these observations, hypotheses and multiple experiments, guidelines are provided to help the designer. Further studies are needed to support and possibly expand these guidelines.

Chapter 6

6 Concluding Remarks

The focus of this research was communicated in Chapter 2 in the form of research questions:

1. Can design variants be evolved using an affordance-genetic algorithm integration that uses end-users' input?
2. Can relationships between affordances and design parameters be extracted from design evolution experiments results?
   - Can affordance and design parameters relationships be used to predict user assessments?

These questions are addressed in the following sections.

6.1 Research Question 1

Answering this question involved the creation of a platform, the Affordance Based Interactive Genetic Algorithm (ABIGA), which integrates the concept of affordances with optimization techniques. The tool is used to extract the input of the end-user in the form of affordance quality evaluations when they're shown design solutions. The results in chapter 4 show that the form of products can be evolved toward solutions that are perceived better by end-users. This is true even when the input from multiple users is used. The ABD/IGA integration has proven to be effective in evolving product shape, that is, the external characteristics of products. How the affordance quality assessments affect the internal aspects of products has not been explored. Chapter 5 showed the importance of knowing how to select product affordances and design parameters to increase the success of product evolution experiments. Though the evolution was tested on two different products, the results suggest that this approach can be applied to a wide variety of products.

6.2 Research Question 2

The form dependency of affordances suggests that the physical characteristics of products are related to affordances. More specifically, the form of products can be related to the quality assessments of their affordances. The results in Chapter 4 and Chapter 5 proved that quality assessments of affordances can be related to the design parameters of products. The fact that these relationships can be expressed mathematically means that the user assessments of the affordances can be predicted when applied to the redesign of the same product or similar products. The logistic regression provided such mathematical models which have the design parameter values as inputs, meaning the designer is able to predict how specific design parameter values will be perceived based on statistical evidence.

6.3 Contributions

The list of contributions to the engineering design community includes

- A platform that integrates Affordance Based Design with Genetic Algorithms that extracts the perceptions of users about products to evolve product form.
- A way to allow parallel user evaluations of interactive genetic algorithm solutions.
- A way to predict how design parameter values affect the perception of end-users toward products' affordance qualities.
- A platform that can be modified to reach crowds to obtain their affordance quality perceptions and use them to evolve products.

6.4 Future Work

The following are areas that can expand the findings of this research. The first two sections talk about expanding the research presented in this dissertation, the rest of the sections are about improvements that can be made to the current design tool.

6.4.1 Effect of Including User Suggested Affordances on Product Form

Users could perceive additional affordances to the ones that the designer included in an experiment while evaluating solutions. The addition of user suggested affordances during an experiment could affect how the product evolves. This is because affordances are tied to the shape of the product as was shown in Chapter 4, but it is not known how adding one or multiple affordances would affect the overall form of the product.

To achieve this, ABIGA needs to be updated to handle extra objectives. A system that handles user affordance suggestions is needed. The challenge lies in being able to add an objective in an experiment that has already started. Moreover, the addition of an affordance might require a change in the topology of the product, because a product feature might need to be added to fulfill the affordance. This of course might involve starting new experiments with the added affordances and updated product structure.

6.4.2 Use of ABIGA in the Early Stages of the Design Process

ABIGA could be used to extract the perceptions from users early in the design process. Besides helping designers gain insight on how the different product affordances relate to form, the results could also map how those perceptions relate to different product architectures. This could help designers identify the product topology that elicits the most positive perceptions from users, effectively helping designers evaluate different solution concepts.

As presented in Chapter 2, users can also be involved in the early stages of the design process. One of the methods presented in that chapter has users evaluate design concepts, that is, sketches or rough prototypes of the different solution variants that designers have created. ABIGA would require basic embodiments of those concepts so representations can be generated and shown to users. The creation of such capability in ABIGA would be a step towards a more automated design process using ABD, a tool that can be used in multiple stages of the design process.

6.4.3 Designer Interface to Create Product Representations

ABIGA currently requires the creation of the product representation through JavaScript code. Two aspects can be improved with a product creation interface (3-dimensional modelling interface). The first aspect is setting up the experiment; if the designer is able to draw the product, the design parameters could be automatically pulled from the drawing to the GA parameters. This interface could also have the user specify the affordances of the product. The second aspect that would be improved is the quality of the product representation shown to users. The same system that renders the product model could be used in the user evaluating interface. 3-dimensional product representations could improve the evaluation of affordance qualities by users.

6.4.4 Automating the Analysis of Results

Chapter four presented ways in which the results of ABIGA could be analyzed and used. Emphasis was given to how design parameter ranges could be extracted from the affordance/design parameter relationships. These parameters can be used to help designers pick design parameter values based on the perceptions from users. ABIGA could benefit from having a data analysis element that automatically runs statistic operations on the results to give design parameter ranges to the designer.

All these improvements and potential future research paths would make ABIGA closer to an industry ready application. Consumer products companies would greatly benefit from the use of ABIGA to gain a better understanding of their products and users, associating product form and user perceptions. One of the challenges presented in Chapter one was to find a way to access users and process their information and include it in the development process, something that would take a long time if done with the user involvement methods presented in Chapter two. ABIGA deals with this by being deployed in the web, meaning many users could access the application with their phones. Companies could submit design experiments to an ABIGA platform and offer incentives to get a large number of users to evaluate their products' affordances.

Product evolution through affordance evaluations could eventually be used to not only improve the external geometry of products, but also the internal aspects of products. This would offer a tool that can be used in multiple stages of the design process, taking advantage of the concept of affordances and optimization tools to automate most of the product development process.

Appendices

Appendix A MySQL Database Queries for Data Analysis

Below are MySQL commands to query design experiment results from the ABI-GA database. MySQL Workbench was used (Oracle, 2016) to extract the data. Note that the computer's IP address used to pull the data from the database needs to be given access permission from the project's Google Cloud Platform console by including the IP address in the settings of the database instance. The names shown in the code are the same names shown in Figure 3.3, Chapter 3.

MySQL Code to Extract the Affordance Evaluations of All Concepts in an Experiment

```
SELECT Concept.idConcept,Concept.Generation,Result.Value
FROM Result JOIN Concept ON Result.Concept_idConcept = Concept.idConcept
JOIN Affordance ON Result.Affordance_idAffordance = Affordance.idAffordance
JOIN Experiment ON Affordance.Experiment_idExperiment = Experiment.idExperiment
WHERE Concept.Generation >= 1 AND Experiment.idExperiment = 1
ORDER BY idConcept,idResult ASC;
```

MySQL Code to Extract the Design Parameter Values of All Concepts in an Experiment

```
SELECT Concept.idConcept, Concept.Generation, DesignParameters.Value
FROM Concept JOIN DesignParameters ON Concept.idConcept = DesignParame-
ters.Concept_idConcept
JOIN DesignParameterConstraints ON DesignParame-
ters.DesignParameterConstraints_idDesignParameterConstraints = DesignParameterCon-
straints.idDesignParameterConstraints
JOIN Experiment ON DesignParameterConstraints.Experiment_idExperiment = Experi-
ment.idExperiment
```

```
WHERE Concept.Generation >= 1 AND Experiment.idExperiment = 1
ORDER BY idConcept,idDesignParameters ASC;
```

MySQL Code to Extract Results of a Single Affordance of All Concepts in an Experiment

```
SELECT Concept.idConcept,Result.Value
FROM Result JOIN Concept ON Result.Concept_idConcept = Concept.idConcept
JOIN Affordance ON Result.Affordance_idAffordance = Affordance.idAffordance
JOIN Experiment ON Affordance.Experiment_idExperiment = Experiment.idExperiment
WHERE Affordance.idAffordance = 1 AND Experiment.idExperiment = 1
ORDER BY idConcept,idResult ASC;
```

MySQL Code to Extract Values of a Single Design Parameter of All Concepts in an Experiment

```
SELECT Concept.idConcept, DesignParameters.Value
FROM Concept JOIN DesignParameters ON Concept.idConcept = DesignParame-
ters.Concept_idConcept
JOIN DesignParameterConstraints ON DesignParame-
ters.DesignParameterConstraints_idDesignParameterConstraints = DesignParameterCon-
straints.idDesignParameterConstraints
JOIN Experiment ON DesignParameterConstraints.Experiment_idExperiment = Experi-
ment.idExperiment
WHERE DesignParameterConstraints.idDesignParameterConstraints=1 AND Experi-
ment.idExperiment = 1
ORDER BY idConcept,idDesignParameters ASC;
```

Bibliography

Banerjee, A., Quiroz, J. C., & Louis, S. J. (2008). A Model of Creative Design Using Collaborative Interactive Genetic Algorithms. *Design Computing and Cognition*, 397–416.

Bentley, P. (1999). *Evolutionary Design by Computers*. Morgan Kaufmann Publishers, Inc.

Brintrup, A. M., Ramsden, J., Takagi, H., & Tiwari, A. (2008). Ergonomic Chair Design by Fusing Qualitative and Quantitative Criteria Using Interactive Genetic Algorithms. *IEEE Transactions on Evolutionary Computation*, *12*(3), 343–354. http://doi.org/10.1109/TEVC.2007.904343

Brintrup, A. M., Ramsden, J., & Tiwari, A. (2007). An interactive genetic algorithm-based framework for handling qualitative criteria in design optimization. *Computers in Industry*, *58*(3), 279–291. http://doi.org/10.1016/j.compind.2006.06.004

Brintrup, A. M., Takagi, H., Tiwari, A., & Ramsden, J. J. (2006). Evaluation of sequential , multi-objective , and parallel interactive genetic algorithms for multi-objective optimization problems †. *Journal of Biological Physics and Chemistry*, *6*(August), 137–146.

Ciccantelli, S., & Magidson, J. (1993). From Experience: Consumer Idealized Design: Involving Consumers in the Product Development Process. *Journal of Product*

Innovation Management, *10*, 341–347. http://doi.org/10.1177/0192513X12437708

Coghlan, H. (1943). The Evolution of the Axe from Prehistoric to Roman Times. *The Journal of the Royal Anthropological Institute of Great Britain and Ireland*, *73*(1), 27–56.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197. http://doi.org/10.1109/4235.996017

Dixon, J., & Poli, C. (1995). *Engineering Design and Design for Manufacturing: A Structured Approach*. Field Stone Publishers.

Dolan, R. J., & Matthews, J. M. (1993). Maximizing the Utility of Customer Product Testing: Beta Test Design and Management. *The Journal of Product Innovation Management*, *10*, 318–330.

Eclipse. (2014). Eclipse Luna software. Retrieved from https://projects.eclipse.org/releases/luna

Gaffney, E. S., Maier, J., & Fadel, G. (2007). Roles of Function and Affordance in the Evolution of Artifacts. In *International Conference on Engineering Design, ICED* (pp. 1–12).

Galvao, A. B., & Sato, K. (2005). Affordances In Product Architecture: Linking Technical Functions And Users' Task (pp. 1–11).

Gaver, W. (1991). Technology affordances. *ACM*, 79–84.

Gen, M., & Cheng, R. (1997). *Genetic Algorithms and Engineering Design*. (H. Parsaei, Ed.). John Wiley & Sons, Inc.

Gibson, J. J. (1979). The theory of affordances. In *The ecological approach to visual perception*. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization & Machine Learning*. Addison-Wesley Publishing Company, Inc.

Google. (2015a). Google App Engine. Retrieved June 5, 2016, from http://code.google.com/appengine/

Google. (2015b). Using the Google Plugin for Eclipse. Retrieved June 5, 2016, from https://developers.google.com/appengine/docs/java/tools/eclipse

Hall, M., Brown, L., & Chaikin, Y. (2008). *Core Servlets and JavaServer Pages* (Second Edi). Boston, MA: Pearson Education, Inc.

Hartson, R. (2003). Cognitive, physical, sensory, and functional affordances in interaction design. *Behaviour & Information Technology*, *22*(5), 315–338. http://doi.org/10.1080/01449290310001592587

Hazelrigg, G. A. (1996). *Systems engineering: An approach to information-based design*. Prentice Hall. Retrieved from http://books.google.com/books/about/Systems_Engineering.html?id=Un5RAAAAMAAJ&pgis=1

Herstatt, C., & Hippel, E. von. (1992). From Experience: Developing New Product Concepts Via the Lead User Method: A Case Study in an "Low Tech" Field. *Journal of Product Innovation Management*, *9*, 213–221.

Hsu, F. C., & Chen, J.-S. (1999). A study on multi criteria decision making model: interactive genetic algorithms approach. In *IEEE SMC'99 Conference Proceedings.*

1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028) (Vol. 3, pp. 634–639). Ieee. http://doi.org/10.1109/ICSMC.1999.823286

Hu, J., & Fadel, G. (2012). Categorizing affordances for product design. In *ASME IDETC/CIE 2012* (pp. 1–15). Chicago, IL.

Kannengiesser, U., & Gero, J. S. (2010). A process framework of affordances in design. *Design Issues*, *28*(1), 50–62.

Kaulio, M. (1998). Customer , Consumer and User Involvement in Product Development : A Framework and a Review of Selected Methods Customer , consumer and user involvement in product development : A framework and a review of selected methods. *Total Quality Management*, *9*(1), 141–149. http://doi.org/10.1080/0954412989333

Kim, Y. S., Shin, J. W., Kim, S. R., Noh, J. H., & Kim, N. R. (2012). A framework of design for affordances using affordance feature repositories. In *ASME IDETC/CIE 2012* (pp. 1–13). Chicago, IL.

Kukkonen, S., & Lampinen, J. (2005). GDE3: the third evolution step of generalized differential evolution. *2005 IEEE Congress on Evolutionary Computation*, *1*, 443–450. http://doi.org/10.1109/CEC.2005.1554717

Loh, W. L. (1996). On latin hypercube sampling. *Annals of Statistics*, *24*(5), 2058–2080. http://doi.org/10.1214/aos/1069362310

Maier, J., & Fadel, G. (2003). Affordance-Based Methods for Design. In *ASME IDETC/CIE 2003* (pp. 1–10).

Maier, J., & Fadel, G. (2007). The affordance structure matrix - A concept exploration and attention directing tool for affordance based design. In *ASME IDETC/CIE 2007* (pp. 1–11). Las Vegas, Nevada.

Maier, J., & Fadel, G. (2009a). Affordance based design: a relational theory for design. *Research in Engineering Design*, *20*(1), 13–27. http://doi.org/10.1007/s00163-008-0060-3

Maier, J., & Fadel, G. (2009b). Affordance-based design methods for innovative design, redesign and reverse engineering. *Research in Engineering Design*, *20*(4), 225–239. http://doi.org/10.1007/s00163-009-0064-7

Maier, J., Sandel, J., & Fadel, G. (2008). Extending the Affordance Structure Matrix – Mapping Design Structure and Requirements To Behavior. *International Design Structure Matrix Conference, DSM*, (November), 6–8.

Mata, I., Fadel, G., & Mocko, G. (2015). Toward automating affordance-based design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *29*(3), 297–305. http://doi.org/10.1017/S0890060415000256

McGrenere, J., & Ho, W. (2000). Affordances : Clarifying and Evolving a Concept. In *Graphics Interface* (pp. 1–8). Montreal.

Minitab. (2016). Minitab. Retrieved January 1, 2016, from https://www.minitab.com/en-us/

Moore, W. L. (1982). Concept testing. *Journal of Business Research*, *10*, 279–294. http://doi.org/10.1016/B978-0-240-81179-6.00018-3

Nguyen, M. T., Guarneri, P., Fadel, G., & Mata, I. (2012). Genetic algorithms applied to affordance based design. In *ASME IDETC/CIE 2012* (pp. 1–7). Chicago, IL.

Norman, D. A. (1988). *The design of everyday things*. New York: Currency Doubleday.

Oracle. (2016). MySQL Workbench. Retrieved April 5, 2016, from https://www.mysql.com/products/workbench/

Orbay, G., Fu, L., & Kara, L. B. (2015). Deciphering the Influence of Product Shape on Consumer Judgments Through Geometric Abstraction. *Journal of Mechanical Design*, *137*(8), 081103–081103. http://doi.org/10.1115/1.4030206

Orsborn, S., & Cagan, J. (2009). Multiagent Shape Grammar Implementation: Automatically Generating Form Concepts According to a Preference Function. *Journal of Mechanical Design*, *131*(12), 121007. http://doi.org/10.1115/1.4000449

Orsborn, S., Cagan, J., & Boatwright, P. (2009). Quantifying Aesthetic Form Preference in a Utility Function. *Journal of Mechanical Design*, *131*, 61001. http://doi.org/10.1115/1.3116260

Otto, K. N., & Wood, K. L. (1998). Product Evolution: A Reverse Engineering and Redesign Methodology. *Research in Engineering Design*, *10*(4), 226–243. http://doi.org/10.1007/s001639870003

Pahl, G., Beitz, W., Feldhusen, J., & Grote, K. H. (2006). *Engineering design: A systematic approach*. Springer.

Park, G.-J. (2007). *Analytic Methods for Design Practice* (1st ed.). Springer-Verlag London. http://doi.org/10.1007/978-1-84628-473-1

Pols, A. J. K. (2012). Characterising affordances: The descriptions-of-affordances-model. *Design Studies*, *33*(2), 113–125. http://doi.org/10.1016/j.destud.2011.07.007

Ramanujan, D., Vinayak, Nawal, Y., Reid, T., & Ramani, K. (2015). Informing Early Design Via Crowd-Based Co-creation. In *ASME IDETC/CIE 2015* (pp. 1–10). Boston, MA.

Raubal, M., & Moratz, R. (2008). *A functional model for affordance-based agents*. *Towards Affordance Based Robot Control* (Vol. 4760). Springer Berlin Heidelberg. http://doi.org/10.1007/978-3-540-77915-5_7

Reid, T., Frischknecht, B. D., & Papalambros, P. Y. (2012). Perceptual Attributes in Product Design: Fuel Economy and Silhouette-Based Perceived Environmental Friendliness Tradeoffs in Automotive Vehicle Design. *Journal of Mechanical Design*, *134*(4), 41006. http://doi.org/10.1115/1.4006146

Reid, T., Gonzalez, R. D., & Papalambros, P. Y. (2010). Quantification of Perceived Environmental Friendliness for Vehicle Silhouette Design. *Journal of Mechanical Design*, *132*(10), 101010. http://doi.org/10.1115/1.4002290

Reid, T., MacDonald, E., & Du, P. (2012). Impact of Product Design Representation on Customer Judgment. *DETC2012-70734, Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, *135*(9), 1–14. http://doi.org/10.1115/DETC2012-70734

Ren, Y., & Papalambros, P. Y. (2011). A Design Preference Elicitation Query as an Optimization Process. *Journal of Mechanical Design*, *133*(11), 111004. http://doi.org/10.1115/1.4005104

Rosenblad-Wallin, E. (1985). User-oriented Product Development Applied to Functional Clothing Design. *Applied Ergonomics*, *16*(4), 279–287. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/8822306

Suh, N. P. (1998). Axiomatic Design Theory for Systems. *Research in Engineering Design*, *10*(4), 189–209. http://doi.org/10.1007/s001639870001

Takagi, H. (1998). Interactive evolutionary computation: System optimization based on human subjective evaluation, 17–19. http://doi.org/10.1109/5.949485

Takagi, H. (2001). Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation. In *Proceedings of the IEEE* (Vol. 89, pp. 1275–1296). http://doi.org/10.1109/5.949485

Tiwari, S., Fadel, G., & Deb, K. (2011). AMGA2 : improving the performance of the archive-based micro-genetic algorithm for multi-objective optimization. *Engineering Optimization*, *43*(4), 37–41.

Tiwari, S., Fadel, G., Koch, P., & Deb, K. (2008). AMGA : An Archive-based Micro Genetic Algorithm for. In *GECCO* (pp. 729–736).

Wolff, J. S., & Wogalter, M. S. (1998). Comprehension of Pictorial Symbols: Effects of Context and Test Method. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *40*(2), 173–186. http://doi.org/10.1518/001872098779480433