JOURNAL OF DIGITAL SOCIAL RESEARCH

WWW.JDSR.IO ISSN: 2003-1998

VOL. 2, NO. 3, 2020, 126–157

ARTIFICIAL INTELLIGENCE AND VIDEO GAME CREATION: A FRAMEWORK FOR THE NEW LOGIC OF AUTONOMOUS DESIGN

Stefan Seidel[a], Nicholas Berente[b], Aron Lindberg[c], Kalle Lyytinen[d], Benoit Martinez[e], and Jeffrey V. Nickerson[c]

ABSTRACT

Autonomous, intelligent tools are reshaping all sorts of work practices, including innovative design work. These tools generate outcomes with little or no user intervention and produce designs of unprecedented complexity and originality, ushering profound changes to how organizations will design and innovate in future. In this paper, we formulate conceptual foundations to analyze the impact of autonomous design tools on design work. We proceed in two steps. First, we conceptualize autonomous design tools as 'rational' agents which will participate in the design process. We show that such agency can be realized through two separate approaches of information processing: symbolic and connectionist. Second, we adopt control theory to unpack the relationships between the autonomous design tools, human actors involved in the design, and the environment in which the tools operate. The proposed conceptual framework lays a foundation for studying the new kind of material agency of autonomous design tools in organizational contexts. We illustrate the analytical value of the proposed framework by drawing on two examples from the development of Ubisoft's *Ghost Recon Wildlands* video game, which relied on such tools. We conclude this essay by constructing a tentative research agenda for the research into autonomous design tools and design work.

Keywords: autonomous design tools, artificial intelligence, organizing, design, innovation, digital innovation, control, work

[a] University of Liechtenstein, Liechtenstein
[b] University of Notre Dame, Indiana, USA
[c] Stevens Institute of Technology, New Jersey, USA
[d] Case Western Reserve University, Ohio, USA
[e] Ubisoft Paris, France

1 INTRODUCTION

Digital technologies increasingly shape the environments in which they operate (Baskerville, Myers, & Yoo, 2019; Rai, Constantinides, & Sarker, 2019) by acting as "performative material devices" (Pickering, 1995). Performativity implies that digital technologies operate with some level of autonomy. Advanced forms of such technologies possess some form of artificial intelligence (AI). Such technologies have information processing capabilities for transforming some inputs into outputs in a way that can be deemed intelligent without close human monitoring. As a result, they have the genuine "capacity … to act on their own, apart from human intervention" (Leonardi, 2011, p. 148). Such autonomy is now evident in a growing array of technologies, including self-driving cars (Badue et al., 2019), conversational agents or chatbots (Cassell, Sullivan, Churchill, & Prevost, 2000), Internet of Things (IoT) applications such as smart homes (Porter & Heppelmann, 2014), and indeed autonomous design tools (Shaker, Togelius, & Nelson, 2016). These technologies can, to an extent, act on their own with little or no human intervention and in ways that are not fully predictable or understandable by humans. These tools also shape their environment in multifaceted ways. Therefore, it is no longer appropriate to view these technologies as passive inert entities to be enacted by humans as controllable tools.

This development has been fueled, in part, by the increased use of AI techniques, such as machine learning or genetic algorithms. These techniques have been evolving for decades in the AI community but have only recently become more widespread and productive in organizational settings (Daugherty & Wilson, 2018). The increased deployment of such autonomous tools has been fueled by effective access to large swaths of data and computing power enabled by the emergence of broadband networks, sensor technologies, cloud-based computing, and platform induced ecosystems (Parker, Van Alstyne, & Jiang, 2016; Tiwana, 2015). As a result, many digital applications are no longer merely passive tools that support or control manual tasks and related organizational processes. They are no longer systems that merely automate a pre-defined process and then 'informate' that process (Zuboff, 1988; Seidel & Berente 2020). In addition, many systems can now act in ways that were previously reserved for human agents (Lyytinen, Nickerson, & King, 2020; Seidel, Berente, Lindberg, Nickerson, & Lyytinen, 2019). This shift has given rise to new concepts, typologies, and notions, such as machines as teammates (Seeber et al., 2019), human-machine-learning (Seidel, Berente, Lindberg, et al., 2019), role-reversal (Demetis & Lee, 2017), digital agency (Ågerfalk, 2020), and meta-human systems (Lyytinen et al., 2020). Humans now delegate tasks to tools that act with autonomy (Ågerfalk, 2020; Zhang, Yoo, Lyytinen,

& Lindberg, forthcoming). Autonomously acting intelligent, learning algorithms increasingly make decisions and engage in value judgements (Baskerville et al., 2019). They rely on their own percepts instead of just executing upon prior knowledge conveyed by their designers (Russel & Norvig, 2016). In some situations, autonomous systems can be conceived of as "users" of humans, rather than the reverse (Baskerville et al., 2019; Demetis & Lee, 2017; Lyytinen et al., 2020). These developments call upon a posthumanist lens that does not identify humans as the sole sources of agency, but considers human and material agencies on equal footing and in symbiotic relationships with a circumscribed sociotechnical system (Latour, 2005; Pickering, 1993, 1995).

One domain that has openly embraced software with autonomous capabilities and epitomizes processes that have traditionally been viewed as human-centric is that of design. Designers across industries increasingly use software-based systems that make independent design decisions. In some cases, these systems execute entire design processes to generate artifacts of ever greater complexity (Seidel, Berente, Lindberg, et al., 2019). Such autonomous design tools employ multiple computational approaches to generate design artifacts, including path-finding algorithms, meta-heuristics (in particular, evolutionary algorithms), and neural networks. Using such techniques, autonomous design tools can now generate a growing variety of multifaceted design artifacts, for instance, nearly full designs of next-generation computer chips (Brown & Linden, 2011; Zhang et al., forthcoming), user interfaces (Yumer, Asente, Mech, & Kara, 2015), three-dimensional virtual worlds (Smelik, Tutenel, de Kraker, & Bidarra, 2010b), and static as well as dynamic content for video games and feature films (Hendrikx, Meijer, Van Der Velden, & Iosup, 2013; Togelius, Yannakakis, Stanley, & Browne, 2011). The applications for such systems are now expanding to mechanical engineering, aerospace, and architecture, among others.

Empirical evidence suggests that autonomous design tools are fundamentally changing the organizing of innovative design work and the way that designers[1] will generate artifacts in the future (Seidel et al., 2018; Zhang et al., forthcoming). Instead of creating artifacts by directly manipulating multifaceted design representations, designers will increasingly focus on selecting system goals, features, and constraints, deciding on related design parameters, setting values for these parameters, and evaluating and learning from the analysis of the tool outcomes (Seidel, Berente, Lindberg, et al., 2019; Seidel et al., 2018; Summerville et al., 2018). Design work in such environments requires designers to be mindful of the

[1] Note that we use the term "designer" in its broadest sense, to refer to engineers, developers, architects, etc., that draw on their expertise to generate solutions.

logic, capabilities, and limitations of the deployed algorithms and to find ways to make sense of and deal with complex and unanticipated outputs. This opens up important questions related to organizing design, including problems of coordination, control and learning in design teams (Puranam, Alexy, & Reitzig, 2014; Seidel, Berente, Lindberg, et al., 2019).

Design automation—autonomous or otherwise—has significantly improved the efficiency of design across a variety of fields. One could easily conceive autonomous design tools simply as the next wave of automation. Indeed, the literature on the algorithms that generate artifacts often highlights the significant potential of these tools to automate design and introduce scale efficiencies (e.g., Smelik, Tutenel, de Kraker, & Bidarra, 2010a; Togelius et al., 2011). This is a reasonable position—designers use the tools first to automate parts of current design practices by carrying out algorithmically specific, relatively complex pre-programmed tasks (such as wiring between gates in chip design). However, these tools will increasingly also make design decisions that are, at least partially, independent of and not fully knowable to the designer. In other words, the tools become black boxed and start acting autonomously. They carry out many tasks with unprecedented speed, scale, and scope so that these activities are likely to materially change the way designers generate artifacts (Summerville et al., 2018). They also exhibit capacities that fundamentally differ from past computer aided design (CAD) tools supporting manual design activities of architects and engineers (Chang & Wysk, 1997; Gupta, Garg, & Chadha, 1981).

Against this backdrop we posit that the use of autonomous tools will continue to generate profound changes in how organizations design, innovate, and organize related activities. **The aim of this paper, therefore, is to formulate a conceptual framework that facilitates future inquiries into how the new and changed material agency of autonomous design tools shapes organizational contexts, how these tools interact with their environment, and how their deployment is likely to lead to novel design processes and artifacts.** To this end, we first conceptualize autonomous design tools a 'rational agents' (Russel & Norvig, 2016) with an embedded design model realized through two separate approaches of information processing: symbolic and connectionist. In a second step, we draw on control theory (Mesarovic, Macko, & Takahara, 1970) to spell out the relationships between autonomous design tools, human designers, and the environment in which the tools are used. At this, we highlight how the need for delegation as well as the frame problem (Dennett, 2006; McCarthy & Hayes, 1981) provide explanations for why control units such as human designers are necessary in typical design situations. We illustrate the analytic value of our model by using two examples adopted from the production of a complex video game software—Ubisoft's *Ghost Recon*

Wildlands. We summarize how autonomous design tools are likely to change the organization of design work in many walks of design given the access to new types of human-machine configurations that are now emerging. We also note avenues for future research on autonomous design tools.

2 AUTONOMOUS DESIGN TOOLS

2.1 From Manual Design to Autonomous Design

Design, in the most general sense of the word, involves the formulation of desirable future states in the world (Goel & Pirolli, 1992). To design is to devise "courses of action aimed at changing existing situations into preferred ones" (Simon, 1996, p. 111). Design is simultaneously mental and representational (Baxter & Berente, 2010; Gero, 1990; Goel & Pirolli, 1992). As result, design processes synthesize a solution by iteratively mobilizing and integrating diverse knowledge elements into varied representations of a solution. Design involves exploration and decomposition, as well as synthesis (Goel & Pirolli, 1992). The outcome of design is an artifact—an object generated by human ingenuity and meeting the goal of changing the given situation to a preferred one.

We can broadly classify approaches to design by their utilization of technologies with increasing degrees of autonomy (Figure 1). At one end of the continuum one can find manual design practices where human designers handcraft artifacts. This does not exclude the use of tools that digitize these practices—drawing tools and CAD tools are prominent examples. Here, tools provide detailed affordances for potential actions (Markus & Silver, 2008; Zammuto, Griffith, Majchrzak, Dougherty, & Faraj, 2007) that can be enacted by designers to manipulate and make sense of the representations. Designers are viewed as craftspeople, who, through their deep knowledge of materials, tools, and design principles, intentionally design and shape an artifact (Sennett, 2008). The notion of affordance describes how these tools become involved in design as they express the meaning and intent of the designer to use a tool feature to achieve a specific goal. In design situations affordances are the action potentials that the material properties of a tool offer to some designer or a group of designers (Markus & Silver, 2008). Designers *enact*—that is, they recurrently interact with the technology (Orlikowski, 2000) in their design practices by putting the technology to use; the enacted affordances improve the design performance of the designers who control the tools.

The more technology starts making decisions on behalf of the designer, the more we can conceive the technology as *acting autonomously*.

If one uses technology that makes autonomous decisions, but still involves intermittent interactions with designers, a hybrid human-machine design system is formed. In such a system the degrees of interaction between autonomous systems and human designers will vary significantly. At a minimum, designers state design requirements (goals, constraints) and complete the design by evaluating it against set up performance goals. The focus is still mainly on automating a specific design task such as a placement, composition, or an optimization problem (Summerville et al., 2018; Togelius et al., 2011). At the other end of the continuum we can find fully autonomous tools that create artifacts without a designer's intervention. This is the case where a machine-learned system independently makes all design decisions and can even adjudicate and establish new design goals (Summerville et al., 2018).

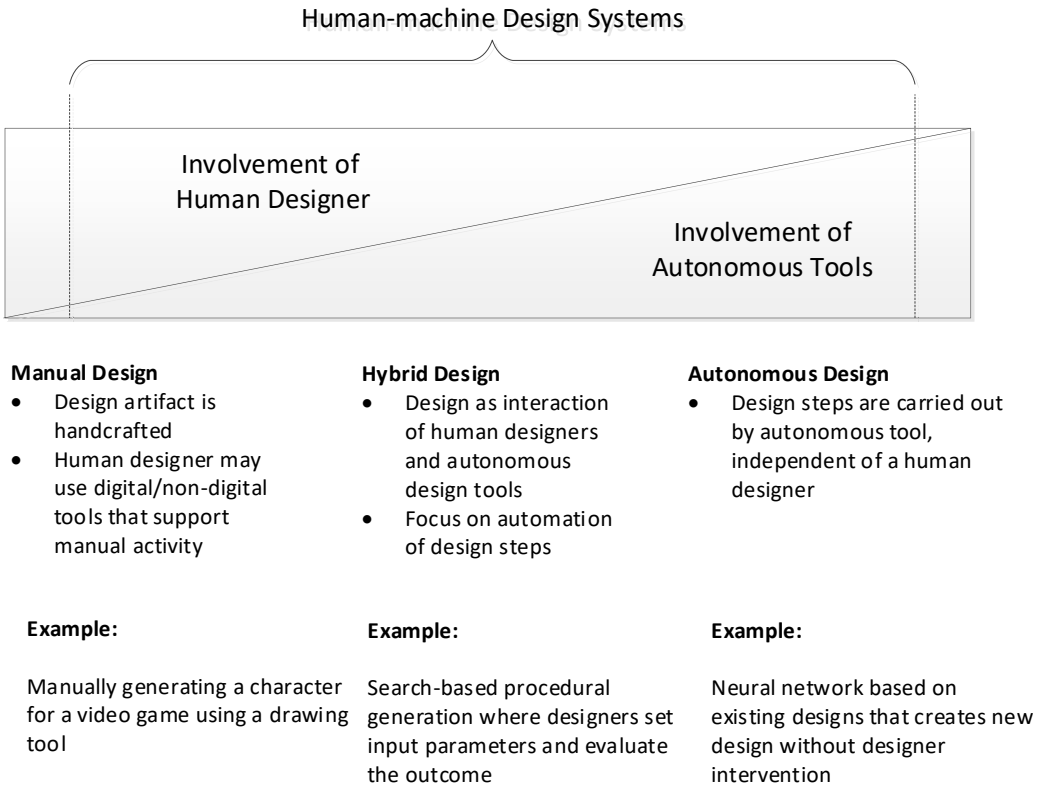

Figure 1. The continuum of human-machine design systems (extended from Seidel, Berente, Lindberg, et al., 2019)

While manual design has dominated all areas of design—from arts and architecture, to engineering—we now see an increased deployment of hybrid human-design systems, where design practices involve rich and multifaceted interactions between designers and varied and complex tool sets. Systems used in design having varying degrees of autonomy have been discussed under multiple labels, including procedural generation

(Ashlock & McGuinness, 2013; Hendrikx et al., 2013), procedural modeling (Müller, Wonka, Haegler, Ulmer, & Van Gool, 2006; Parish & Müller, 2001), computational creativity (Liapis, Yannakakis, & Togelius, 2014), generative design systems (Krish, 2011), and autonomous generation (Summerville et al., 2018). What these tools share in common is that they (partially) replace manual craftsmanship in that they generate design artifacts with relatively infrequent designer intervention to find novel solutions that meet given goals and constraints. In this view, autonomous design tools are machine-based agents that perform design work alongside human agents. **Autonomous design tools are software tools that, once started, independently make design decisions to generate design outcomes based on varied forms of input and using an embedded, often complex, unknown, and evolving design model.**

2.2 The Key Elements of Autonomous Design Tools

Autonomous design tools, as defined above, are rational agents. Russel and Norvig (2016) describe rational agents as entities that perceive their environment through sensors, act upon the environment through actuators, and whose behavior can be described in terms of an agent function. In addition, there needs to be some performance measure by which the success of the agent's actions can be evaluated. Rational agents act autonomously to the extent that they rely on their own percepts (the input they receive from the environment) and less on the prior knowledge of their designers (Russel & Norvig, 2016)

We can thus define autonomous design tools by their inputs, their outputs, and, in between, the computational process underlying the specific design decisions the systems make. Embedded design models broadly determine the ways in which the tool will generate outcomes based on a set of input parameters (Seidel, Berente, Lindberg, et al., 2019). That is, as with other information technologies, these tools link inputs to outputs through some form of information processing. The key, however, is that this information processing allows the tool to generate a design outcome by making design decisions that are at least partially independent from human designers (or, more broadly, the users of the tool). From the perspective of the designer interacting with the tool, these outputs can also be unpredictable and surprising. While the designer may have a broad understanding of what the tool is expected to do, he or she cannot precisely anticipate what the tool will produce given the inputs. This is different from mere automation, where a given task is accomplished through a deterministic, traceable process and the designer knows what the output will be given his or her inputs. Consequently, the outcomes generated by

autonomous tools are often perceived as being creative by humans (Boden, 2009; Veale, Cardoso, & y Pérez, 2019).

Drawing on Russel and Norvig's (2016) conceptualization of rational agents that have information processing capacity and interact with their environment by receiving sensory input and acting upon that environment, Figure 2 delineates an abstract model of an autonomous design tool. In this view, an autonomous design tool receives sensory input from the environment, makes design decisions based on an embedded design model, and then generates some output that adds content to, or alters, existing design content. Note that the embedded design model can be implemented in various ways, ranging from a simple reflex agent to a learning agent that involves a learning element which allows making improvements based on the model's interaction with the environment (Russel & Norvig, 2016). We next turn to two dominant approaches to implementing embedded design models.

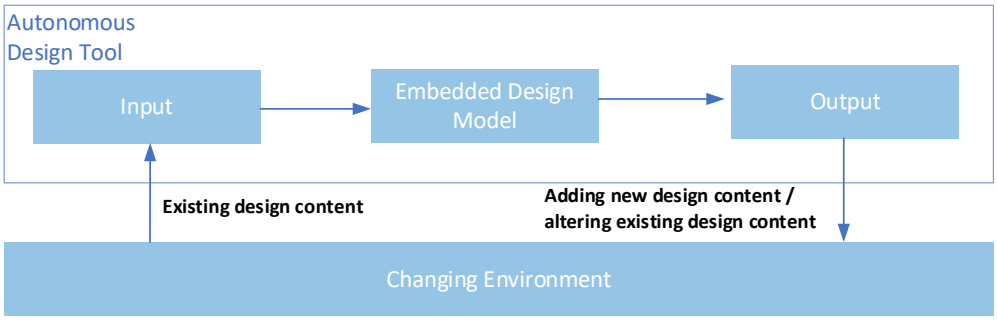

Figure 2. Autonomous tool interacting with its environment (adapted from Russel & Norvig, 2016)

2.3 Two Dominant Types of Embedded Design Models

There are two dominant approaches to the implementation of autonomous design tools—physical symbol systems and non-symbolic connectionist systems. They are based on two main perspectives of information processing (Smolensky, 1987; Sun, 1999). The first is founded on the explicit manipulation of symbol systems expressing the embedded design model based on formal logic. This approach presupposes that the features to be manipulated have already been identified and that consequences of its manipulation can be largely predicted. The second non-symbolic approach is founded on the implicit feature discovery facilitated by connectionist systems, epitomized by artificial neural networks. We can apply these two approaches to distinguish between two broad types of embedded design models (i.e., the computational models that define how the tool works) of autonomous design tools (Table 1). Note that the symbolic vs. non-symbolic

categorization is typically maintained in order to distinguish two types of applications in artificial intelligence (e.g. Sun, 1999).

Table 1. Two types of embedded design models

Type of embedded design model	Description	Autonomous design example
Physical symbol system	• Approaches based on explicit representations and symbolic programming • Transformation of physical symbols based on rules • The rules represent a designer's understanding of how the autonomous design tools should address its design approach • Explicit representation of the problem space in terms of relevant features	Search-based algorithms such as pathfinding (Pohl, 1970) in procedural game development (Togelius et al., 2011) Rule-based procedural content generation (Smith & Mateas, 2011)
Non-symbolic/ connectionist systems	• Implicit representation of the problem as the systems discover variables, correlations between variables, and correlations between correlations • Transformation of inputs and outputs through a multilayered network • The underlying representational model is opaque to the designer • No explicit conceptual foundation • Typically based on large data sets ("big data")	Neural network used to reduce complex design problems as in the case of designing user interfaces at Adobe Labs (Yumer et al., 2015) Adversarial networks to generate visual content based on models trained on existent designs (Summerville et al., 2018) Terrain design through adversarial neural networks trained on real-world terrain as well as their sketched counterparts (Guérin et al., 2017)

2.3.1 The Physical Symbol Systems Approach

The physical symbol systems approach is based on the premise that the sort of problem-solving associated with design work is essentially about transforming symbol structures until a result is reached that is satisfactory by some performance measures (Newell & Simon, 1972). In such situations, designers—human or non-human—search a large, multi-dimensional, potentially unbounded, problem space to identify a solution. A problem space is comprised of an initial state, a goal state, and a set of operators that allow a movement from the initial state to the goal state (Newell & Simon, 1972). Designers need a representation of the problem space in order to make it possible to apply operators (Simon, 1996): "A problem

representation structures the problem space with elements of the problem and its potential solution and is the most potent explanation for if, and how, a design problem will be solved" (Boland, 2004, p. 106). Throughout the design process, the designer generates design representations that are tested against his or her cognitive schemata for goal satisfaction (Baxter & Berente, 2010). Hence, design can be understood as a search built on nested generate-test cycles that seeks satisfactory solutions for a given, often changing and fluid, design problem (Buchanan, 1992). In this view, optimization is possible, but only in formally constrained and well-structured design situations such as optimal placement of logic gates on a relatively small semiconductor chip, where, for example, optimization techniques such as dynamic programming can be applied. Optimization, however, is a distant or impossible goal in most real-world design situations. The problem spaces are simply too large and complex, and the search takes too much time and effort. While the actual problem space (Dorst & Cross, 2001; Newell & Simon, 1972; Simon, 1996) might be known in an abstract sense (such as in chip design), it is impossible to explore all feasible solutions. Therefore, designers need to employ satisficing procedures and rely on heuristics such as means-ends analysis that involve recursively decomposing the problem into subcomponents until concrete operators can be applied to find solutions that are acceptable rather than optimal (Simon, 1996). Oftentimes design in complex situations adopts procedures that produce initial conditions for further design (Simon, 1996). These procedures can result in a series of component optimization and satisficing actions that ultimately constitute the outcome of the design process.

Typical applications of this approach are based on search-based (Togelius et al., 2011) and rule-based (Smith & Mateas, 2011) approaches for generating design artifacts. Search-based algorithms generate alternative solutions step-by-step and evaluate them. Rule-based approaches apply a set of rules to derive a satisfactory outcome. Tools using these approaches generate large scale artifacts as these tools, upon receiving input from the designers, can normally undertake multiple design steps independently (Ashlock & McGuinness, 2013; Hendrikx et al., 2013).

A prominent application area of these types of design tools is in the procedural generation of content for video games (Ashlock & McGuinness, 2013; Hendrikx et al., 2013). Such content generation can happen at build time (before the game is shipped) and runtime (when the player has started the game). Procedural generation is, in contrast to manual content production, "the application of computers to generate game content, distinguish interesting instances among the ones generated, and select entertaining instances on behalf of the players" (Hendrikx et al., 2013, p. 1:2). Recent well-known examples of the use of build-time procedural

content generation can be found in open-world games were users can freely explore a vast virtual environment. These games are based on the availability of large game spaces that would be prohibitively expensive to create without the help of systems that generate large parts of the space without much human designer intervention.

2.3.2 The Connectionist Approach

Connectionist approaches, most notably artificial neural networks, provide an alternative. This approach is now used in multiple fields, including design applications (e.g., Yumer et al., 2015). These approaches do not need pre-existing ontologies or features. Instead, such systems discover features from raw sensory data. That is, they do not need pre-existing "theories" and "constructs" to operate; they will discover variables, correlations between variables, and correlations between correlations by themselves. Neural networks can extend the abstraction of such processes layer-by-layer until higher-level constructs in data are discovered, capturing real-world features such as objects, words, and sentences. Because of the mechanisms through which such networks operate, they also are good at compressing information in efficient ways and reducing the dimensionality of large datasets. Through such reduction, design problems can be made more amenable for human designers to navigate a limited set of critical parameters. A key difference from search- or rule-based approaches, which generate content through searching a design space, is that these tools directly generate content (Summerville et al., 2018) in that the systems are trained on successful or representative designs and then can generate other, similar designs (Summerville et al., 2018). In this approach it is not necessary to codify explicit design knowledge in terms of search algorithms that can generate content and then evaluate that content; embedded design models based on connectionist approaches are therefore an important step towards increasing autonomy as they do not rely on the prior knowledge of their designers (Russel & Norvig, 2016).

In the case of designing interfaces at Adobe, for instance, designers were confronted with a problem space that was too large for human designers to navigate—approximately 100 parameters controlled processes for generating navigation structures (Yumer et al., 2015). They turned to creating a deep neural network that helped them to reduce the high-dimensional space to a three-dimensional space that designers could control through slider bars. The designers describe how they used a learning system instead of a rule-based, procedural modeling system to tackle the high dimensionality of the problem as follows:

> Procedural modeling systems allow users to create high quality content through parametric, conditional or stochastic rule sets. While such

> approaches create an abstraction layer by freeing the user from direct geometry editing, the nonlinear nature and the high number of parameters associated with such design spaces result in arduous modeling experiences for non-expert users. We propose a method to enable intuitive exploration of such high dimensional procedural modeling spaces within a lower dimensional space learned through autoencoder network training (Yumer et al., 2015, p. 109).

Symbolic and connectionist approaches for generating design outcomes can also be combined. For instance, we can conceive of search-based algorithms that generate outcomes but where the evaluation occurs through a trained neural network (Summerville et al., 2018).

3 THE CONTEXT OF AUTONOMOUS DESIGN TOOLS: A CONTROL PERSPECTIVE

The continuum from pure manual design to fully autonomous design highlights that autonomous design tools will operate in relation to multiple elements involved in the design process—human designers, autonomous design tools, and the environment. Fully autonomous design tools that define the design problem and devise solutions are a distant goal, and we need to consider these tools from a socio-technical perspective where human and machine designers interact synergistically. There are at least two reasons that require a human agent in such design systems. First, from an operational perspective, human designers *delegate* a design task to an autonomous tool, set parameters, start the autonomous design tool, and evaluate the outcome and make adjustments to the set of input parameters. Second, considering that problem spaces are evolving and that the same tool might be used for different design situations (and hence problem spaces), autonomous design tools suffer from the *frame problem*, which describes how algorithms are constrained by the rules (i.e., the knowledge) they currently possess and are hence incapable of reacting to environment states for which they are not prepared (Dennett, 2006; McCarthy & Hayes, 1981; Salovaara, Lyytinen, & Penttinen, 2019). In the case of symbolic approaches, the frame problem would demand that rules are added to the embedded design model to make it applicable to a broader or changing set of design problems (Dennett, 2006). However, even if we assume that we can infinitely add rules, such approach will increase the system's complexity and render its performance useless. While connectionist approaches involve learning, they still suffer from the frame problem as they are typically solving "closed-world" problems and remain constrained by the specific goal functions and available data (Salovaara et al., 2019).

Therefore, in this section, we turn to the interaction between autonomous design tools and their control units, most notably human

designers, in relation to the environment in which they operate. We apply a control perspective (Mesarovic et al., 1970) to express the morphology of autonomous design tools. Figure 3, which is an extension of Figure 2, highlights the principal relationships of an autonomous design tool with its control system and environment.

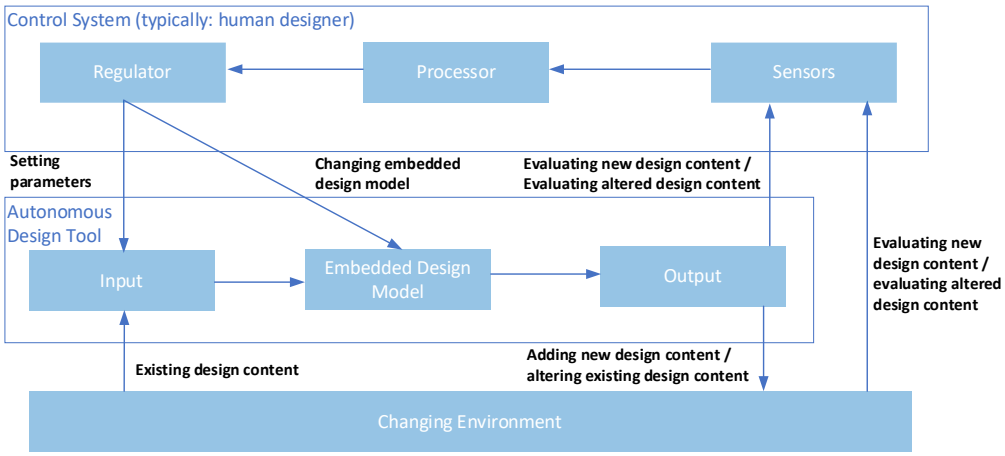

Figure 3. Control system, (partially) autonomous design tool, and the changing environment

The environment embodies the material and social context within which the design system as a whole operates and with which it interacts. It is likely to be changed through the use of the autonomous tool and its outcomes. The tool's input function refers to the interfaces through which the tool receives information about the environment. The embedded design model refers to the procedures followed to process information from the inputs, implemented through symbolic approaches, connectionist approaches, or a combination of such approaches. The output function represents the mechanisms through with the system effectuates the results of this process on the environment. An autonomous design tool is never entirely independent and its environment involves a second system—a control system—which triggers the autonomous design tool, monitors and evaluates its performance, and may even change the embedded design model in order to react to alterations in the problem space, thereby addressing the frame problem. On this view, model evolution can result from both the model's ability to learn and the intervention of the control unit changing the embedded design model (Seidel, Berente, Lindberg, et al., 2019). We describe the three components—control system, the autonomous design tool itself, and the environment—in what follows.

The human designer or design team, as a control system, involves three aspects: sensors, processors, and regulators. Sensors involve the designer's or design team's perception of the output of the autonomous

tool. Of note is that while current applications typically involve human designers who work together with tools to create content (Seidel et al., 2018; Smelik et al., 2010a; Summerville et al., 2018) we can also think of non-human control systems and even autonomous design tools as control systems. However, as indicated earlier, addressing the frame problem will eventually require a human agent who is able to change the model to react to changes in the problem space. This creates a hierarchy of nested systems of human designers and autonomous design tools. This can involve the monitoring of performance measures applied to the design alternatives. Processing involves the interpretation and analysis of that sensory information to assess adequacy and the degree to which design preferences have been met. Regulators are the ways that designers change conditions of design activity. This could include modifying parameters or changing the design of the system as well as modifying or implementing a new algorithm.

The autonomous design tools receive sensory input through two channels: (1) through parameters specified by the designers programming or guiding the tool, that is, through the regulating component of the control system; (2) through input they receive from their interactions with the environment. Systems that are based on the symbol system approach, for instance, transform one symbol structure (e.g., an already existent representation of the design artifact) into another symbol structure (i.e., the new representation of the design artifact). The tool can make multiple design decisions without the intervention of the control unit. Eventually, however, some result will be evaluated by the control unit which may lead to new input and additional iterative cycles of deploying the tool. We describe the impact of an autonomous design tool on its environment in terms of the tool's output function. This design outcome might be a stand-alone artifact (e.g., a layout of a semi-conductor chip) or embedded artifact (e.g., modifications to a landscape in a video game, for instance, through adding a road network).

Finally, the environment is the context in which the autonomous tool operates and which it changes. The environment provides sensory inputs to the autonomous design tool. A search-based algorithm might receive a three-dimensional landscape as input and then generate alterations of this landscape until the process terminates with a satisfactory solution (Seidel et al., 2018). Similarly, a machine-learned model might be fed with a partial design and then complete that design (Summerville et al., 2018). Ultimately, whether or not the design outcome is satisfactory depends on how well it performs in the environment in which it is deployed. The environment can include both social (e.g., human stakeholders who have a say in whether an artifact meets the expectations) and technical (e.g., requirements of other components when designing more complex systems) elements. Table 2

provides an overview of how autonomous design tools, through their input and output functions, interact with control systems as well as the environment.

Table 2. Key components of autonomous design tools and their relationships to control systems and the environment

Component	Definition	Example
Input function (sensory)	Autonomous design tools receive input • from the designer guiding the autonomous tool (i.e., the regulating component of the control system) and • from the design environment.	The designer of a semiconductor chip sets parameters such as component parameters, physical parameters, and electrical parameters.
Embedded design model	The embedded design model—the algorithms and data models—determine how the tool designs; variants include: • models based on physical symbolic systems; • machine-learned models based using non-symbolic systems; • hybrids.	Can range from heuristics to machine-learned algorithms and may even involve a number of cooperating algorithms
Output function (actions)	The output function describes the actual actions that the tools takes with regards to its environment. The output function together with the embedded design model represent the actuating element of the autonomous tool as a goal-seeking system.	Autonomous design tools generate artifacts or change existing artifacts, for instance, the layout of a semiconductor chip.

Based on this conceptualization, we can further identify three key dimensions to characterize autonomous design tools: autonomy, interactivity, and understandability. First, the extent to which the autonomous tool requires pre-defined rules (either built-in or set by the control system such as a human designer) defines the level of autonomy. As indicated earlier, the less design tools depend on the prior knowledge of their designers (Russel & Norvig, 2016) the more autonomous they are.

Second, we can distinguish two types of interactivity: interactivity with the control system and interactivity with the environment. The more input is required from the regulator as part of the control system, the more interactive the design process is in terms of control-system-autonomous-tool interaction. Moreover, the autonomous tool may receive sensory input from the changing environment; the more input the tool receives from the environment that informs its course of action the more interactive the design process is in terms of tool-environment interaction.

Third, from the viewpoint of the human designer acting as the control unit, autonomous tools can exhibit different levels of understandability—while the functioning of a pathfinding algorithm for generating road networks may be relatively easily comprehended (and thus how the tool generated a result), this will be different in the case of neural networks training an artificial intelligence—reflecting recent discussions on explainability of artificial intelligence (Miller, 2018; Samek, Wiegand, & Müller, 2017).

Table 3 provides an overview these properties.

Table 3. Key properties of designer-autonomous-design-tools-systems

Property	Description	Example
Level of autonomy	Autonomous tools can rely on inputs provided by the designers (e.g., parameters) as well as information they receive from their interaction with the environment, i.e., with the problem space.	Chip design tools generate entire sections of a chip without direct intervention of the human designer.
Level of interactivity	While autonomous design tools can perform design activities with little to no user intervention, this does not mean that they operate in isolation. There are two types of interactivity: (1) Interactivity with the environment: the tool receives sensory input from the environment and acts upon this input, in turn changing the environment and generating new sensory input. (2) Interactivity with the control system: The tool receives input from the control system, be it a human designer or another tool.	The designer in the production of an asset (e.g., a landscape) for a video game monitors the process of the autonomous tool and, based on intermediate results, changes input parameters.
Level of understandability	The embedded design model of an autonomous design tool might be more or less easy to understand for the designer—or might be very complex.	Semiconductor chip designers cannot predict how the tool will layout components and also cannot always make sense of why particular design decisions were made by the tool.

4 ILLUSTRATIVE EXAMPLES: CONTENT GENERATION IN VIDEO GAMES

Autonomous design tools are now widely used to produce content for a new generation of video games. Current tools focus on procedural generation and mainly rely on symbolic approaches to identifying satisficing solutions. However, there are also some examples of learning algorithms, for instance: algorithms for terrain generation are trained on real-world terrains (Guérin et al., 2017). While tools make design decisions independently from the human designer, there is still significant interaction with human designers (Seidel, Berente, Lindberg, et al., 2019). Such algorithmically generated content can include a variety of game elements—including textures, buildings, road networks, etc. Designers typically combine these elements with specific hand-crafted elements. The interplay of automated and manual generation of content is crucial as humans are looking for rich and unique experiences, and undirected automated generation might lead to results that are not perceived as being authentic.

Ubisoft's *Ghost Recon Wildlands*, an action adventure game, is a recent example where designers used autonomous tools to generate large parts of the game space (Seidel, Berente, Lindberg, et al., 2019). Guided by human designers, algorithms procedurally generated much of the background content, and designers then tweaked what algorithms created and further handcrafted elements in the game space. In this process the tools would, for instance, generate large amounts of a detailed terrain. Then the designers would modify the terrain further and add extra detail. Some areas of the game space were still generated in a manual fashion. This combined process required developing and selecting appropriate tools and models that would align with the core concepts of the game as specified by a team of designers and developers. Next, we consider two examples from *Ghost Recon Wildlands* and interpret these examples through our conceptual lens.

The first example is the generation of a road network using a pathfinding algorithm.[2] The path finding algorithm transforms a data structure (a landscape without a road) into a different data structure (a landscape with a road). While the road itself is generated by the algorithm, this case is still characterized by interaction between the human designer—who acts as a control system—and the autonomous design tool. The human designer sets parameters (such as start and end points), runs the system, evaluates the outcome, and runs the tool again, until there is a satisfactory result. Importantly human designers are also involved in developing and selecting the specific algorithm and hence the design model embedded in the tool. Figure 4 highlights how different algorithms produce quite

[2] The process described here was inspired by Galin, Peytavie, Guérin, and Beneš (2011).

different designs. This illustrates how the selection of the algorithm—and hence the model embedded in the tool—is essential for the design outcome. Notably, this design outcome provides key input for further design steps which again involve the use of autonomous design tools, including for the generation of fences, crash barriers, traffic signs, road markings, specific types of grass or rocks on the roadside, powerlines along roads, etc. This indicates how the design outcomes generated by autonomous design tools fundamentally impact on the design process, including subsequent design decisions that both other autonomous design tools and human designers make.

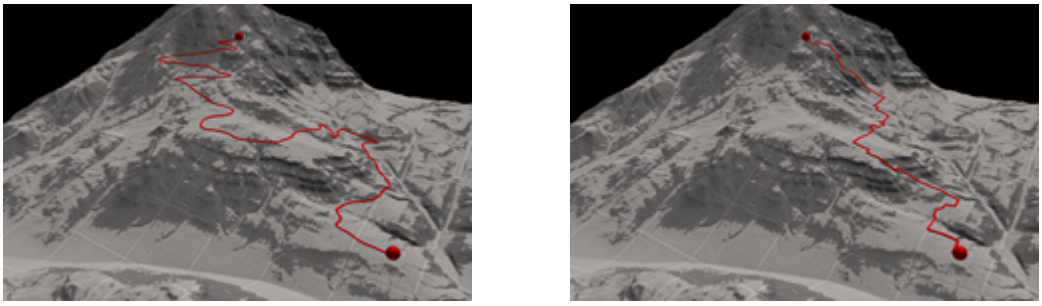

Figure 4. Generation of a road network using different algorithms (Source: Ubisoft)

In this example, all key components of an autonomous design tool and their context are present (Table 4). First, the tool receives sensory input (the topology of the map). Second, the tool computes a solution, in this case using a search-based algorithm, without much user intervention. Third, the tool acts upon the environment by adding the road network to the landscape, thereby altering the design artifact. Figure 5 shows an example of the output.

Figure 5. Autonomously generated road (Source: Ubisoft)

Table 4. Example: Generation of a road network in a videogame

Component		Example
Control system		The control system is a human designer using the tool to generate roads/a road network for a video game. The tool is executed by the designer and then evaluated by the human designer. The human designer must thus possess knowledge of the underlying embedded design model to anticipate what the algorithm does.
Autonomous design tool	Input function	Designer's specification in terms of start and end points Existing design content in terms of the landscape in which the road network is placed, e.g., the road can only have a certain incline otherwise an alternative path needs to be taken
	Embedded design model	Pathfinding algorithm The design tool searches the problem space by devising design alternatives
	Output function	Coordinates of the road network that fit to the landscape Alteration of design artifact, resulting in a landscape with road network
Environment		Altered design artifact: roads connecting start and end-points in the game space

Our second example, the generation of villages in the game space, allows us to further highlight how autonomous design tools and human designers interact (Figure 6).[3] This process starts with key decisions made by the human designer, including the identification of a center point for a village/town and the identification of related areas. These are key decisions that impact the road pattern within the town, and we can describe this process as a form of "architectural structuring" (Seidel, Berente, & Gibbs, 2019). The actual buildings are then placed by a self-aware packing algorithm. This process unfolds without human intervention and is based on building definitions, each of which has their own placement rules. Still, human designers have at their disposal tools to tweak what the algorithm has designed. The key is that placing the buildings involves design decisions that are made by a tool. Figure 6 displays the definition of a center for a village (I), the definition of the village boundaries (II), the definition of internal paths and zones (III), and the process of placing buildings (IV)—it is this stage where the design tool takes over.

[3] The process described here was inspired by Emilien, Bernhardt, Peytavie, Cani, and Galin (2012).

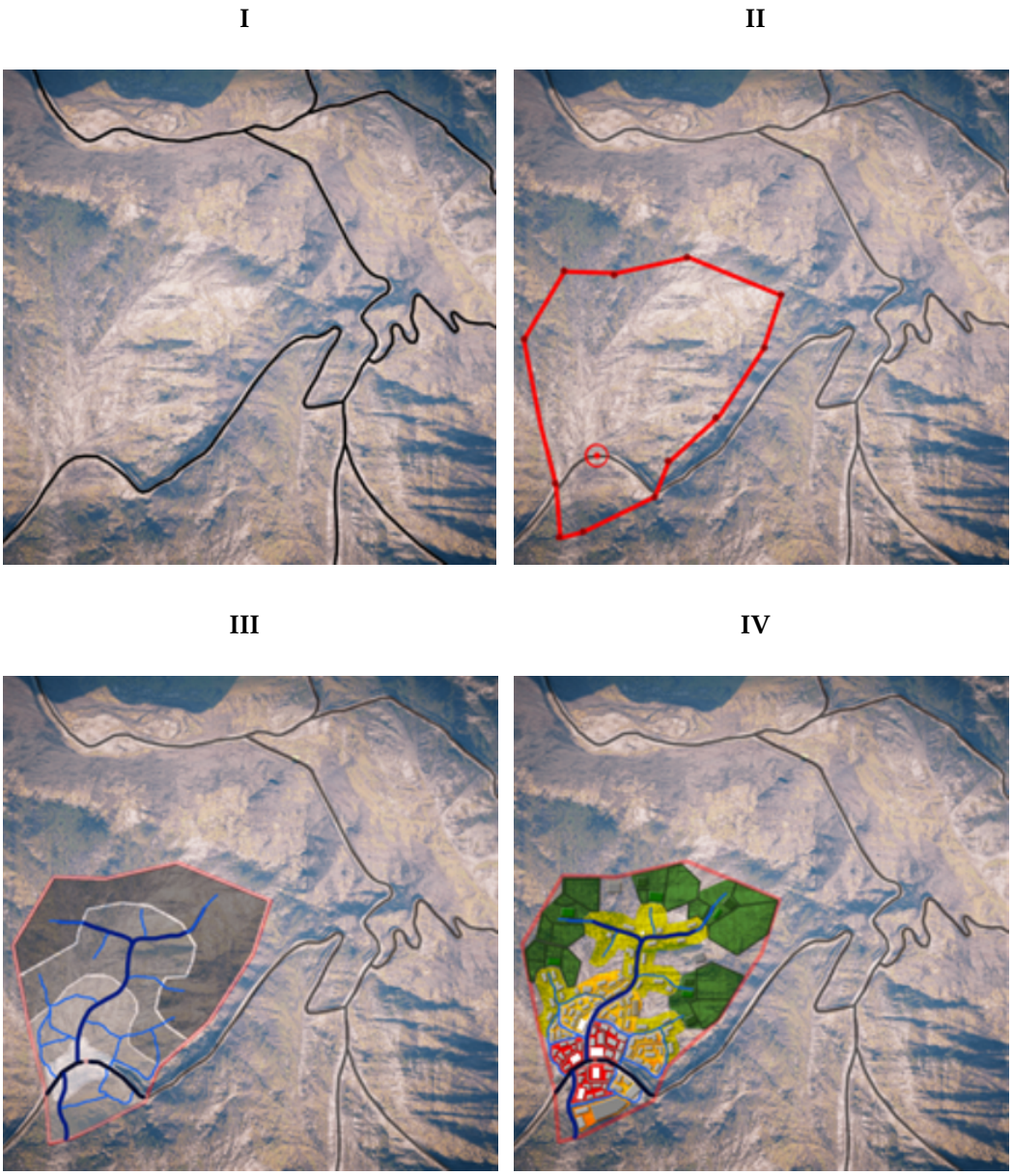

Figure 6. Steps in generating villages using a self-aware packing algorithm (Source: Ubisoft)

In this case all elements of autonomous design tools are present. The human designer acts as the control system and provides key inputs to the tool—such as the identification of areas to focus on. The autonomous design tool has an embedded design model in terms of a self-aware packing algorithm and the tool generates output that alters the environment in which the tool operates. Table 5 provides an overview and Figure 7 shows an example of a village generated using this approach.

Figure 7. Village generated through interaction of human designer and autonomous design tools (Source: Ubisoft)

Table 5. Example: generation of villages in a videogame

Component		**Example**
Control system		The control system is a human designer who makes key architectural decisions: • Location of the village • Type of pattern (radial or square) • Internal road structure • Zoning • Decision on the buildings to use, including placement definitions for each building Each step is manually triggered so the user can visually validate the result before adjusting parameter of the current step or move to the next one. The human designer must thus possess knowledge of the underlying embedded design models to anticipate what the algorithms do.
Autonomous design tool	Input function	The center point of the village and boundaries A specific set of parameters for the different functions can be saved as a preset and reused elsewhere. Producing a different (for a different location) but predictable result in term of pattern and layout
	Embedded design model	Space partitioning Pathfinding Self-aware recursive packing algorithm
	Output function	Alteration of design artifact, resulting in a terraformed landscape with the village footprint

		Trajectories representing internal roads and paths that are used for further detailing through placing objects (lamppost, signs, etc.) on roadside 3d models of buildings
Environment		Altered design artifact: villages with dedicated center, roads, and other elements are added to the game space

With regards to the key properties of autonomy, interactivity, and understandability the two examples are comparable. First, the tools in these examples make design decisions on behalf of their designers, who however still have to provide user input. They can thus be described as being partially autonomous. There is interactivity as after setting parameters and running the tools again, the designers may still alter the resulting artifacts. This interactivity becomes particularly visible in the staged process of designing villages that moves from identifying a location to the actual placement of buildings and that involves interdependent designer and tool decisions. Finally, the embedded design models—such as the pathfinding algorithm and the self-aware packing algorithm—are quite understandable for the designers who use these tools.

5 DISCUSSION: A RESEARCH AGENDA FOR AUTONOMOUS DESIGN TOOLS AND CHANGING DESIGN WORK

Our key intention with this article is to provide a conceptual framework for studying the interactions between human and machine components in design systems that involve autonomous design tools, and therefore enabling theorizing of the materiality of autonomous design tools in relation to the organizing of design work. The literature on autonomous design tools (such as procedural generation) has so far largely focused on the technical aspects of implementing these approaches. Still, some scholars have indicated that these tools need to be considered in concert with the human designers employing such tools (Seidel, Berente, Lindberg, et al., 2019; Smelik et al., 2010b; Summerville et al., 2018). Hence, a socio-technical perspective on design tools becomes increasingly important as scholars have started to revisit expanded notions of material agency in the presence of increasingly autonomous and intelligent systems by using labels such as human-machine-learning (Seidel, Berente, Lindberg, et al., 2019), role-reversal (Demetis & Lee, 2017), digital agency (Ågerfalk, 2020), or meta-human systems (Lyytinen et al., 2020). Our conceptualization of autonomous design tools based on a rational agent perspective and control theory highlights how designing with autonomous tools is a process that is co-constituted by the activities of human designers and the design activities carried out by autonomous design tools. We have suggested that human

designers act as control systems that "coach" autonomous design tools which act in a partially independent fashion in the sense that they make design decisions that cannot necessarily be anticipated by the human designers running the tools. However, despite increasing levels of system autonomy, humans still play a pivotal role as a control unit for the autonomous design tool.

The autonomous design tools we have discussed in this paper are tools designed for specific tasks. When Newell, Shaw, and Simon described their general problem solver (Newell, Shaw, & Simon, 1959), they conceived of a more general approach of computational problem solving based on the use of general heuristics of means-end-analysis and planning. Such a general approach to design still seems to be a distant goal. However, we have highlighted some developments in this direction such as using adversarial neural networks (Guérin et al., 2017) that foreshadow a development towards more flexible autonomous design tools. Following from this analysis of specificity and generality of tools, one key question is about the extent to which we can expect to find regularities in the way designers and machines interact when carrying out different tasks, and hence about the limits of theories about these new forms of human-machine interaction.

Against this background, autonomous design tools pose a variety of novel research challenges that recognize the socio-technical nature of designing with such tools. Here, we categorize these challenges into four areas to offer a systematic research agenda that can encourage interdisciplinary research teams to pursue fruitful and innovative research programs in this nascent field (Table 6).

Table 6. Research agenda

Phenomenon / level of analysis	**Example research questions**
Designer-autonomous-tool-interaction	How do humans and autonomous design tools interact *effectively* in design processes? How can the outcomes of using autonomous design tools be evaluated under different conditions; how to address the cognitive overload of human designers? How does learning take place when humans and autonomous tools interact? What forms of interaction and processes lead to better learning outcomes and design outcomes? How is such hybrid learning different from pure cognitive models of experiential learning or crafting? How do designers work with different types of embedded design models? What are the differences between interacting

	with design systems that are based on symbolic approaches versus those that are based on connectionist approaches?
Organizing design work with autonomous tools	How do autonomous design tools change designer roles, interactions, design principles, and organizing? How do key organizational processes such as decision making and sensemaking unfold in situations where human designers interact with autonomous design tools? How do organizational practices evolve as autonomous tools are introduced to design settings? Do organizational tasks or domains matter for how autonomous design tools are used and integrated?
Autonomous tools and markets/crowds/communities	How does the use of autonomous design tools change labor markets? Can autonomous design tools emerge as market-based agents that carry out specific design tasks and be offered as a service?
Ethical considerations of using autonomous design tools	What are the ethical dimensions and implications of using autonomous design tools? Are there regulatory issues related to recording and justifying design decisions and outcomes carried out by autonomous tools?

5.1 Designer-Autonomous Tool-Interaction

One important aspect that differentiates autonomous design tools from other types of software systems is that they generate outcomes where the human designer often cannot foresee the specifics of the outcome (Seidel, Berente, & Gibbs, 2019; Zhang et al., forthcoming). This is possible because these tools act autonomously as they move through the process of generating or altering an artifact while making invisible design decisions that do not depend on their designer's (the one who designed the tool) prior knowledge of the design task as they go. Still, the designer makes initial assumptions about the design setting and goals (choosing tools, choosing parameters, setting parameters). This then yields contextual information (mostly about the design artifacts) which helps this designer to further guide the tool. Autonomous tools, through their independent design decisions, generate information that informs computations going forward, as well as the designer's subsequent actions.

These observations indicate that we need to attend to the specific ways in which designers and tools engage with each other. It seems warranted to move our attention from the idea of designers enacting technology to processes of mutual enactment, where human activity and machine activity constitute each other in situ. However, as contemporary design tools such as those used in video game production still require designer input, we can ascribe a certain head status to the human designer (Leonardi, 2011). Still,

we can conceive of a future where the boundaries of control and controlled will increasingly vanish, perhaps requiring a more symmetrical conceptualization of the relationship between control and controlled. There is no reason to believe that control in human-machine design systems could not reside in a machine or could be shared among human designers and autonomous design tools. The move from "technology enactment" to "mutual enactment" requires us to explore the specific ways designers interact with their tools and how they do so effectively. Moreover, we can expect that there will be new challenges in evaluating the outcomes generated by tools as well as through the interaction of designers and tools. Finally, it will be interesting to see how the nature of the embedded design model (symbolic versus connectionist or any combination) impacts on the interaction between human designers and tools.

5.2 Organizing Design Work with Autonomous Tools

It is likely that the increased use of autonomous design tools will involve moving away from an understanding of the designer as a craftsman (Sennett, 2008), towards being a tool chauffer. Designers increasingly need to develop a generalized understanding of the design problem as well as the envisioned solution so that they can think about appropriate strategies to generate design outcomes (which manifests in the selection and configuration of tools, including the selection of the embedded design model), instead of actively generating the design artifact through dedicated, manual design activities where each step is evaluated against the design goal. This requires us to rethink the way that we conceive of the institutional role of a designer, as it has potential implications for the way that education and learning will change across fields of practice.

In light of this changing role of the designer in relation to their materials and tools, it will further be important to explore if and how key organizational processes such as decision making related to participating or producing (March & Simon, 1993) as well as sensemaking (Weick, 1995; Weick, Sutcliffe, & Obstfeld, 2005) change in situations where autonomous tools become part of the fabric of organizing. Sensemaking, for instance, has been conceptualized as a retrospective process where not only cognition impacts action but where action impacts cognition (Weick, 2001)—but what does it mean for human cognition if this action is performed on their behalf by a machine with potentially unpredictable outcomes?

Finally, organizations want to understand the specific outcomes generated by autonomous design tools and how they fit into the overall product and service portfolio. While autonomous design tools promise to offload repetitive work from designers and quickly generate design artifacts of unprecedented scale with comparably little resources, it is also

clear that these tools have limitations. While their output is complex and often unanticipated, they are still largely deterministic systems. The question arises to which extent these tools can indeed be creative in the sense that they generate truly novel artifacts; there is a risk that the tools will generate repetitive, perhaps boring (Backus, 2017) and non-creative content. Still, it seems reasonable to make two claims regarding the creativity involved while using these tools. First, if we conceive of autonomous design tools as part of a socio-technical design system where two components (humans and machines) interact, and where the output of each element impacts the action of the other, the overall system acts creatively, if it generates outcomes that are both novel and useful (Amabile, 1996) and that would not have been produced without such interactions. Second, the outcomes that are generated by the types of autonomous design tools we described in this paper exhibit a complexity that makes them unpredictable, and thus potentially novel, from the designers' point of view (Seidel, Berente, & Gibbs, 2019).

5.3 Autonomous Tools and Markets/Crowds/Communities

The described changes in the way humans and machines interact as well as in the way we organize for work can be seen as micro foundations for broader level changes at multiple levels of analysis. We may, for instance, expect that the labor market will, going forward, require different designer skills. Specifically, designers will require in-depth knowledge about how to select, orchestrate, and run autonomous design tools. Moreover, software development skills will be important for designers as they seek to understand and perhaps alter the models embedded in autonomous design tools.

Moreover, it will be interesting to see to what extent autonomous design tools will not only be used to create products, but also function as market-based agents that offer services. In the past, software-as-a-service and related concepts have mainly focused on providing capabilities such as for data storage and process automation. If autonomous design tools become market-based agents that carry out design tasks on behalf of a customer, organizations will rely on external stakeholders to perform design work. This bears the potential for disrupting a variety of industries, as the generation and implementation of purposeful design outcomes is a key source of value generation in many contemporary organizations. What, however, would the consequences be if such tasks could be performed at higher speed, higher scale, and perhaps decreased cost by an external provider?

5.4 Ethical Considerations of Using Autonomous Design Tools

Finally, we have to attend to the ethical dimensions and implications of using autonomous design tools. For instance, these tools deeply penetrate into the types of work that have traditionally be seen to be reserved for humans—work that is related to creativity and design. We can thus expect that these tools will challenge established role identities of designers and related professions and that may even lead to situations where designers feel threatened by that technology (Seeber et al., 2020). Following from this observation, it is crucial to explore the pertinent regulatory issues related to recording and justifying design decisions and outcomes carried out by autonomous tools. This involves questions with regards to the intellectual property that is generated by autonomous design tools as well as the consequences of using such intellectual property.

6 CONCLUSION

In this paper we have discussed the conceptual foundations of autonomous design tools. These foundations prepare the ground to study how these tools are involved in socio-technical systems and how they change how we organize design work. To this end, we have highlighted how designers currently have tools at their disposal that range from tools which provide limited support for manual tasks, to design tools which are fully autonomous. Moreover, we have argued that the idea of fully autonomous design tools remains an abstraction; the practical examples we have identified in areas such as the design of video games, which formed the baseline example in this paper, rely on the interaction of human designers and tools. We also distinguished two general approaches to building autonomous design tools (physical symbol systems and connectionist systems) and we have highlighted how there is now a nascent interest in tools that learn from interactions with their environment, thus moving us closer to the vision of fully autonomous design tools.

After having experienced two AI winters, AI and associated design systems are finally flourishing. These developments have been driven by vast amounts of available data upon which machine learning algorithms are capitalizing, as well as the emergence of cloud-based computing infrastructures that provide the necessary fuel, the computing power necessary to explore vast design spaces. The emergence of these technologies heralds a possible revolution in how we think about design across multiple domains. It is therefore incumbent on us to seek to thoroughly understand this new breed of tools and the consequences of their usage.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under grants 2026583, 1909803, 1717473, and 1745463.

REFERENCES

Ågerfalk, P. J. (2020). Artificial intelligence as digital agency. *European Journal of Information Systems*, 29(1), 1-8.

Amabile, T. M. (1996). *Creativity in context*: Westview Press.

Ashlock, D., & McGuinness, C. (2013). Landscape automata for search based procedural content generation. In *2013 IEEE Conference on Computational Intelligence in Games (CIG).*

Backus, K. (2017). Managing output: Boredom versus chaos. In T. X. Short & T. Adams (Eds.), *Procedural Generation in Game Design* (pp. 13-21): AK Peters/CRC Press.

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., . . . Mutz, F. (2019). Self-driving cars: A survey. arXiv preprint arXiv:1901.04407.

Baskerville, R., Myers, M., & Yoo, Y. (2019). Digital first: The ontological reversal and new challenges for IS. *MIS Quarterly, 44(2), 509-523*.

Baxter, R. J., & Berente, N. (2010). The process of embedding new information technology artifacts into innovative design practices. *Information and Organization*, 20(3-4), 133-155.

Boden, M. A. (2009). Computer models of creativity. *AI Magazine*, 30(3), 23-34.

Boland, R. (2004). Design in the punctuation of management action. In R. Boland & F. Collopy (Eds.), *Managing as designing: Creating a vocabulary for management education and research* (106-112). Stanford, California: Stanford Business Books

Brown, C., & Linden, G. (2011). *Chips and change: How crisis reshapes the semiconductor industry*: MIT Press.

Buchanan, R. (1992). Wicked problems in design thinking. *Design Issues*, 8(2), 5-21.

Cassell, J., Sullivan, J., Churchill, E., & Prevost, S. (2000). *Embodied conversational agents*: MIT press.

Chang, T.-C., & Wysk, R. A. (1997). *Computer-aided manufacturing*: Prentice Hall.

Daugherty, P. R., & Wilson, H. J. (2018). *Human + machine: Reimagining work in the age of AI*: Harvard Business Press.

Demetis, D., & Lee, A. (2017). When humans using the IT artifact becomes IT using the human artifact. In *50th Hawaii International Conference on System Sciences*.

Dennett, D. C. (2006). Cognitive wheels: The frame problem of AI. In C. Hookway (Ed.). *Minds, Machines and Evolution* (pp. 129-150): Cambridge University Press.

Dorst, K., & Cross, N. (2001). Creativity in the design process: Co-evolution of problem-solution. *Design Studies*, 22(5), 425–437.

Emilien, A., Bernhardt, A., Peytavie, A., Cani, M.-P., & Galin, E. (2012). Procedural generation of villages on arbitrary terrains. *The Visual Computer*, 28(6-8), 809-818.

Galin, E., Peytavie, A., Guérin, E., & Beneš, B. (2011). Authoring hierarchical road networks. In *Computer Graphics Forum* (Vol. 30, No. 7, pp. 2021-2030). Oxford, UK: Blackwell Publishing Ltd.

Gero, J. S. (1990). Design prototypes: a knowledge representation schema for design. *AI Magazine*, 11(4), 26-36.

Goel, V., & Pirolli, P. (1992). The structure of design problem spaces. Cognitive science, 16(3), 395-429.

Guérin, É., Digne, J., Galin, E., Peytavie, A., Wolf, C., Benes, B., & Martinez, B. (2017). Interactive example-based terrain authoring with conditional generative adversarial networks. *ACM Transactions on Graphics (TOG)*, 36(6), 228.

Gupta, K. C., Garg, R., & Chadha, R. (1981). *Computer aided design of microwave circuits*. NASA STI/Recon Technical Report A, 82.

Hendrikx, M., Meijer, S., Van Der Velden, J., & Iosup, A. (2013). Procedural content generation for games: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 9(1), 1:2-1:24.

Krish, S. (2011). A practical generative design method. *Computer-Aided Design*, 43(1), 88-100.

Latour, B. (2005). *Reassembling the social: An introduction to actor-network-theory*: Oxford University Press.

Leonardi, P. M. (2011). When flexible routines meet flexible technologies: Affordance, constraint, and the imbrication of human and material agencies. *MIS Quarterly*, 35(1), 147-168.

Liapis, A., Yannakakis, G. N., & Togelius, J. (2014). Computational game creativity. In Proceedings of the *5th International Conference on Computational Creativity*.

Lyytinen, K. a., Nickerson, J. V., & King, J. L. (2020). Metahuman systems = humans + machines that learn. *Journal of Information Technology*.

March, J., & Simon, H. A. (1993). *Organizations* (2nd ed.). New York: Wiley.

Markus, M. L., & Silver, M. S. (2008). A foundation for the study of IT effects: A new look at DeSanctis and Poole's concepts of structural features and spirit. *Journal of the Association for Information Systems*, 9(10), 609-632.

McCarthy, J., & Hayes, P. J. (1981). Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer & D. Michie (Eds.), *Machine Intelligence 4* (pp. 463-502). Edinburgh: Edinburgh University Press.

Mesarovic, M. D., Macko, D., & Takahara, Y. (1970). *Theory of hierarchical, multilevel, systems* (Vol. 68). New York: Academic Press.

Miller, T. (2018). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267, 1-38.

Müller, P., Wonka, P., Haegler, S., Ulmer, A., & Van Gool, L. (2006). Procedural modeling of buildings. *ACM Transactions on Graphics (TOG)*, 25(3), 614-623.

Newell, A., Shaw, J. C., & Simon, H. A. (1959). Report on a general problem-solving program. In *IFIP Congress*.

Newell, A., & Simon, H. A. (1972). *Human problem solving* (Vol. 104, No. 9): Prentice-Hall Englewood Cliffs, NJ.

Orlikowski, W. J. (2000). Using technology and constituting structures: A practice lens for studying technology in organizations. *Organization Science*, 11(4), 404-428.

Parish, Y. I., & Müller, P. (2001). Procedural modeling of cities. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*.

Parker, G., Van Alstyne, M., & Jiang, X. (2016). *Platform ecosystems: How developers invert the firm*. Boston University Questrom School of Business Research Paper.

Pickering, A. (1993). The mangle of practice: Agency and emergence in the sociology of science. *American Journal of Sociology*, 99(3), 559-589.

Pickering, A. (1995). *The mangle of practice: Time, agency and science*. Chicago, IL: The University of Chicago Press.

Pohl, I. (1970). Heuristic search viewed as path finding in a graph. *Artificial Intelligence*, 1(3-4), 193-204.

Porter, M. E., & Heppelmann, J. E. (2014). How smart, connected products are transforming competition. *Harvard Business Review*, 92(11), 64-88.

Puranam, P., Alexy, O., & Reitzig, M. (2014). What's "new" about new forms of organizing? *Academy of Management Review*, 39(2), 162-180.

Rai, A., Constantinides, P., & Sarker, S. (2019). Editor's comments: Next-generation digital platforms: Toward human–AI hybrids. *MIS Quarterly*, 43(1), iii-ix.

Russel, S., & Norvig, P. (2016). *Aritifical intelligence. A modern approach*: Pearson.

Salovaara, A., Lyytinen, K., & Penttinen, E. (2019). High reliability in digital organizing: Mindlessness, the frame problem, and digital operations. *MIS Quarterly*, 43(2), 555-578.

Samek, W., Wiegand, T., & Müller, K.-R. (2017). *Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models*. arXiv preprint arXiv:1708.08296.

Seeber, I., Bittner, E., Briggs, R. O., de Vreede, T., de Vreede, G.-J., Elkins, A., . . . Randrup, N. (2019). Machines as teammates: A research agenda on AI in team collaboration. *Information & Management*, 103174.

Seeber, I., Waizenegger, L., Seidel, S., Morana, S., Benbasat, I., & Lowry, P. B. (2020). Collaborating with technology-based autonomous agents. *Internet Research*, 30(1), 1-18.

Seidel, S. and Berente, N. (2020) "Automate, Informate, and generate: Affordance primitives of smart devices and the Internet of Things," in S. Nambisan, K. Lyytinen, & Y. Yoo (Eds.), *Handbook of Digital Innovation*, Northampton: Edward Elgar Publishing.

Seidel, S., Berente, N., & Gibbs, J. (2019). Designing with autonomous tools: Video games, procedural generation, and creativity. In *Proceedings of the 40th International Conference on Information Systems*.

Seidel, S., Berente, N., Lindberg, A., Nickerson, J. V., & Lyytinen, K. (2019). Autonomous tools & design work: A triple-loop approach to human-machine learning. *Communications of the ACM*, 62(1), 50-57.

Seidel, S., Berente, N., Martinez, B., Lindberg, A., Lyytinen, K., & Nickerson, J. V. (2018). Succeeding with autonomous tools in systems design: Reflective Practice & Ubisoft's Ghost Recon Wildlands Project. *IEEE Computer*, 51(10), 16-23.

Sennett, R. (2008). *The craftsman*. London: Allen Lane.

Shaker, N., Togelius, J., & Nelson, M. J. (2016). *Procedural content generation in games*: Springer.

Simon, H. A. (1996). *The sciences of the artificial*. Cambridge, MA: MIT Press.

Smelik, R. M., Tutenel, T., de Kraker, K. J., & Bidarra, R. (2010a). Integrating procedural generation and manual editing of virtual worlds. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*.

Smelik, R. M., Tutenel, T., de Kraker, K. J., & Bidarra, R. (2010b). Interactive creation of virtual worlds using procedural sketching. In *Eurographics (Short papers)* (pp. 29-32).

Smith, A. M., & Mateas, M. (2011). Answer set programming for procedural content generation: A design space approach. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(3), 187-200.

Smolensky, P. (1987). Connectionist AI, symbolic AI, and the brain. *Artificial Intelligence Review*, 1(2), 95-109.

Summerville, A., Snodgrass, S., Guzdial, M., Holmgård, C., Hoover, A. K., Isaksen, A., . . . Togelius, J. (2018). Procedural content generation via machine learning (PCGML). *IEEE Transactions on Games*, 10(3), 257-270.

Sun, R. (1999). Artificial intelligence: Connectionist and symbolic approaches. In N. J. Smelser & P. B. Baltes (Eds.), *International Encyclopedia of the Social & Behavioral Sciences*, 783-789: Elsevier.

Tiwana, A. (2015). Evolutionary competition in platform ecosystems. *Information Systems Research*, 26(2), 266-281.

Togelius, J., Yannakakis, G. N., Stanley, K. O., & Browne, C. (2011). Search-based procedural content generation: A taxonomy and survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(3), 172-186.

Veale, T., Cardoso, F. A., & y Pérez, R. P. (2019). Systematizing creativity: A computational view. In T. Veale & A. Cardoso (Eds.), *Computational Creativity* (pp. 1-19): Springer.

Weick, K. E. (1995). *Sensemaking in organizations* (Vol. 3): Sage.

Weick, K. E. (2001). *Making sense of the organization*. Malden, MA, USA: Blackwell Publishing.

Weick, K. E., Sutcliffe, K. M., & Obstfeld, D. (2005). Organizing and the process of sensemaking. *Organization Science*, 16(4), 409-421.

Yumer, M. E., Asente, P., Mech, R., & Kara, L. B. (2015). Procedural modeling using autoencoder networks. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*.

Zammuto, R. F., Griffith, T. L., Majchrzak, A., Dougherty, D. J., & Faraj, S. (2007). Information technology and the changing fabric of organization. *Organization Science*, 18(5), 749-762.

Zhang, Z., Yoo, Y., Lyytinen, K., & Lindberg, A. (forthcoming). The unknowability of autonomous tools and the liminal experience of their use. *Information Systems Research*.

Zuboff, S. (1988). *In the age of the smart machine: The future of work and power* (Vol. 186): Basic books New York.