Key Competences with Physical Computing

Mareen Przybylla[1], **Ralf Romeike**[2]
[1]Didactics of Computer Science
University of Potsdam
A.-Bebel-Str. 89
14482 Potsdam, Germany
przybyll@cs.uni-potsdam.de
[2]Computing Education Research Group
Friedrich-Alexander-Universität Erlangen-Nürnberg
Martensstraße 3
91058 Erlangen, Germany
ralf.romeike@fau.de

Abstract: Physical computing covers the design and realization of interactive objects and installations and allows students to develop concrete, tangible products of the real world that arise from the learners' imagination. This way, constructionist learning is raised to a level that enables students to gain haptic experience and thereby concretizes the virtual. In this paper the defining characteristics of physical computing are described. Key competences to be gained with physical computing will be identified.

Keywords: Defining characteristics of physical computing, key competences in physical computing, physical computing tools

1 Introduction

Within the last decades computers have become ubiquitous and interactive media. Embedded systems play an increasingly important role in our everyday-lives, but only few students ever get the chance to investigate and understand them, although this persistent trend can be used in computing education. Constructionist learning can be strengthened and inspiring learning environments offered, where students can be creative. One way to address these issues is to implement physical computing in computer science classrooms. Everyone likes making stuff do things – just think of marble slides, domino effects, Rube Goldberg machines or flying paper airplanes. With physical computing such childhood dreams can be met in the classroom while at the same time understanding embedded and ubiquitous media.

This article will give an overview of the defining characteristics of physical computing and describe typical products, processes and tools of the discipline. The question will be pursued, what key competences students gain in physical computing.

2 Three Pillars of Physical Computing

Physical computing is quite a new and unfamiliar concept among informatics teachers and informatics didactic researchers. Literature review has shown that different authors put different emphases on the following three pillars of physical computing: typical products, tools and processes of physical computing need to be investigated in order to clarify the meaning. It will not be possible to clearly draw lines between the three fields. Processes involve tools and aim at particular products, so different perspectives will therefore certainly put a stronger focus on any of the three variables, but not neglect the other two. As a starting point it is assumed that physical computing means to creatively design tangible interactive objects or systems using programmable hardware.

2.1 Products

The term 'physical computing' in educational settings was first mentioned by O'Sullivan and Igoe (2004), who see it as a crucial element of such systems that they make use of transducers (sensors and actuators) to connect the virtual and the physical world. Typical products of physical computing are programmed tangible media, such as the example illustrated in Figure 1. Such media can be embedded, interactive, responsive, adaptive and many more. One feature all products of physical computing have in common is that they are not transformational. Conventional computing systems (e.g. text translation or batch processing) do not allow interference in the processing as they strictly collect input data, then process these data and finish the program by presenting the result. Physical computing devices run continuously and interact steadily with the environment. Siemers (2012) distinguishes interactive from reactive systems depending on the communication driver. In the following both are referred to as interactive objects:

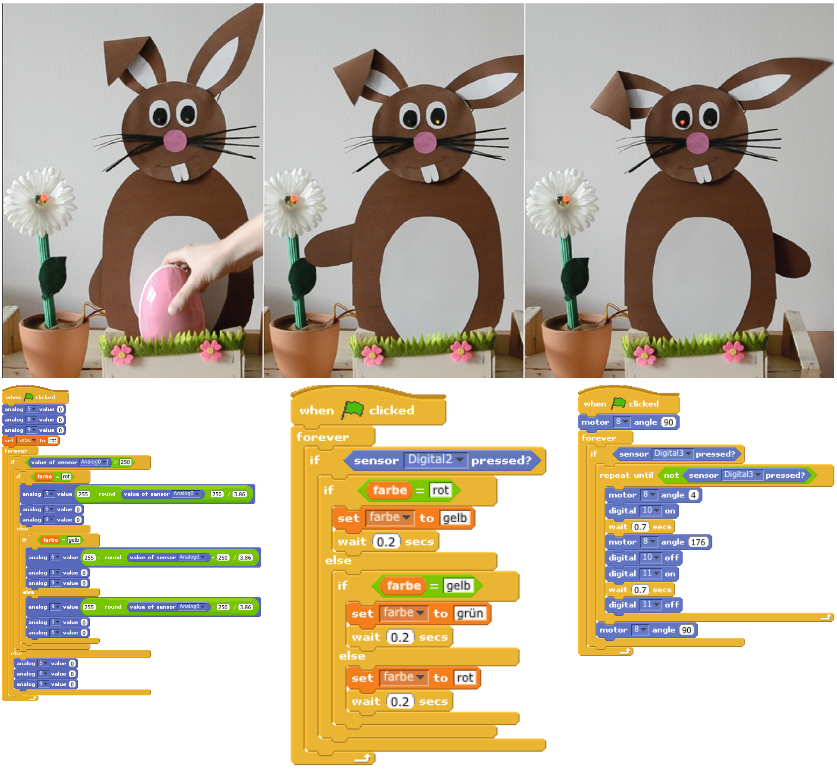
> Interactive Objects are programmed, tangible media containing an integrated system that is invisible to the outside world. They perceive their

> environment with sensors, which in turn deliver data to be processed by the integrated system. According to the configuration of the system these data are passed on to actuators. Interactive objects can be part of networks of interactive installations. (cf. Przybylla, Romeike, 2012)

Examples for such interactive objects and installations range from interactive jewelry and clothes over intelligent toy pets and mood lamps to room-filling installation arts.

2.2 Processes

There are many different contexts in which physical computing is practiced and therefore also a variety of purposes that are pursued. O'Sullivan & Igoe (2004) strengthen the role of the physical body in computing. They encourage makers to forget what they know about computers when planning a new project and instead to focus on the needs of people and the environment that are to be supported by computers. Others have adapted physical computing and use it in a wider meaning: they see it as connecting computers to the physical world (e.g. Computer Laboratory, 2013; Libow Martinez, Stager, 2013). This shifts the focus from interaction of machines with humans (only) to interaction between machines and the physical world in general. Among artists and designers, physical computing is characterized by the use of electronics to prototype new materials. Another aspect is the process of tinkering: reusing and improving existing hard- and software in an experimental way, driven by curiosity, imagination and creativity is part of the process (Banzi, 2011). This is also reflected in many projects, where people have used existing toys or devices to make something new. Based on the suggestions of O'Sullivan and Igoe (2004), the common approach of physical computing can be split into two parts:



```
when ⚑ clicked
analog 5 value 0
analog 6 value 0
analog 9 value 0
set farbe to rot
forever
  if value of sensor Analog0 > 250
    if farbe = rot
      analog 5 value 255 - round value of sensor Analog0 - 250 / 3.86
      analog 6 value 0
      analog 9 value 0
    else
      if farbe = gelb
        analog 6 value 255 - round value of sensor Analog0 - 250 / 3.86
        analog 5 value 0
        analog 9 value 0
      else
        analog 9 value 255 - round value of sensor Analog0 - 250 / 3.86
        analog 5 value 0
        analog 6 value 0
  else
    analog 5 value 0
    analog 6 value 0
    analog 9 value 0
```

```
when ⚑ clicked
forever
  if sensor Digital2 pressed?
    if farbe = rot
      set farbe to gelb
      wait 0.2 secs
    else
      if farbe = gelb
        set farbe to grün
        wait 0.2 secs
      else
        set farbe to rot
        wait 0.2 secs
```

```
when ⚑ clicked
motor 8 angle 90
forever
  if sensor Digital3 pressed?
    repeat until not sensor Digital3 pressed?
      motor 8 angle 4
      digital 10 on
      wait 0.7 secs
      motor 8 angle 176
      digital 10 off
      digital 11 on
      wait 0.7 secs
      digital 11 off
    motor 8 angle 90
```

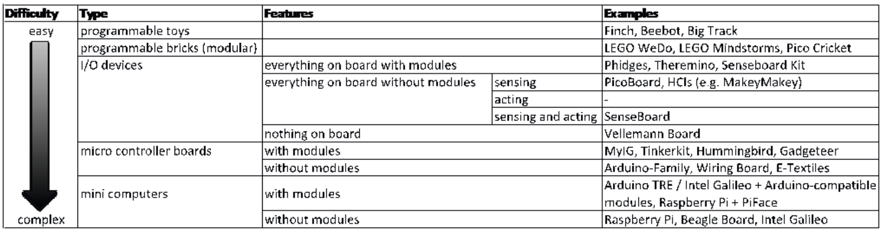
Figure 1: Vigilant Easter Bunny made with My Interactive Garden and S4A

- Description of what is supposed to happen from the point of view of the person who is going to experience it (e.g. “The thief who wants to steal the Easter egg will be chased away from the bunny, which makes noise, moves its arms ...”)
- Description of how this is supposed to happen (e.g. “When the thief takes the egg, this is noticed with a digital infrared sensor placed in the nest...”)

Physical computing thus focuses on ideas, not on technical limitations. This way, whim, imagination and creativity are fostered.

2.3 Tools

By now there is a large variety of good and affordable hardware on the market, which can be used for physical computing. Obviously, physical computing always involves the use of sensors and actuators and a computer in-between to control the behavior. Apart from this restriction, anything is possible. It is therefore not surprising, that tools are available in big numbers for every purpose and target group. Different types of hardware for physical computing activities were classified technically, based on their level of complexity and additional features such as modularity (Figure 2). Five main groups of hardware tools have been identified: programmable toys, I/O devices, programmable bricks, microcontroller boards and mini computers. Depending on how advanced the makers are, they either assemble sensors and actuators themselves or use preassembled hardware. For the above-mentioned “Vigilant Easter Bunny” the physical computing toolkit “My Interactive Garden” consisting of an Arduino microcontroller and plug and play components and the programming environment Scratch for Arduino (S4A) reduce the complexity of using and programming the interactive object. In some primary school contexts, programmable toys and bricks were used to introduce students to algorithmic thinking. For some purposes (e.g. physical measurements) and first interactions input and output devices are helpful. Strictly speaking, the latter cannot be associated with physical computing, because the resulting products cannot be described as interactive objects in the sense of physical computing. Input and output devices, however, can serve as sensor and actuator boards and be connected to the integrated system. This way, they become relevant hardware tools. It is also conceivable that whole laptop or tablet computers are integrated into interactive objects to make use of displays, cameras, et cetera.



Difficulty	Type	Features		Examples
easy	programmable toys			Finch, Beebot, Big Track
↓	programmable bricks (modular)			LEGO WeDo, LEGO Mindstorms, Pico Cricket
	I/O devices	everything on board with modules		Phidges, Theremino, Senseboard Kit
		everything on board without modules	sensing	PicoBoard, HCIs (e.g. MakeyMakey)
			acting	-
			sensing and acting	SenseBoard
		nothing on board		Vellemann Board
	micro controller boards	with modules		MyIG, Tinkerkit, Hummingbird, Gadgeteer
		without modules		Arduino-Family, Wiring Board, E-Textiles
	mini computers	with modules		Arduino TRE / Intel Galileo + Arduino-compatible modules, Raspberry Pi + PiFace
complex		without modules		Raspberry Pi, Beagle Board, Intel Galileo

Figure 2: Classification of suitable tools for physical computing

2.4 Physical Computing in the Computer Science Classroom

Summarizing, physical computing is an activity that involves creative arts and design processes and that, by bringing together hard- and software components, connects the virtual world of computers to the physical world of humans. All hardware components used in physical computing make use of transducers to interact steadily with their environment. Typical tools used for physical computing include microcontrollers and minicomputers. Physical computing projects are of an iterative nature and quickly bring forth working prototypes. In every iteration ideas are always in focus, the implementation of the projects will start only after the ideas have matured (Figure 3). The artistic approach in physical computing matches perfectly with the ideas of constructionism: According to the constructionist learning theory, learning is most effective when learners construct knowledge and develop competences from their own initiative and for a personally relevant purpose (Papert, Harel, 1991). Resnick (1996) added: "What's important is that they are actively engaged in creating something that is meaningful to themselves or to others around them". Further, in all physical computing projects prototypes are created and iterative processes are cycled. Sometimes, existing systems are reused or expanded. In a constructionist sense, learners will hereby develop meaningful products they can present to and discuss with friends and family.

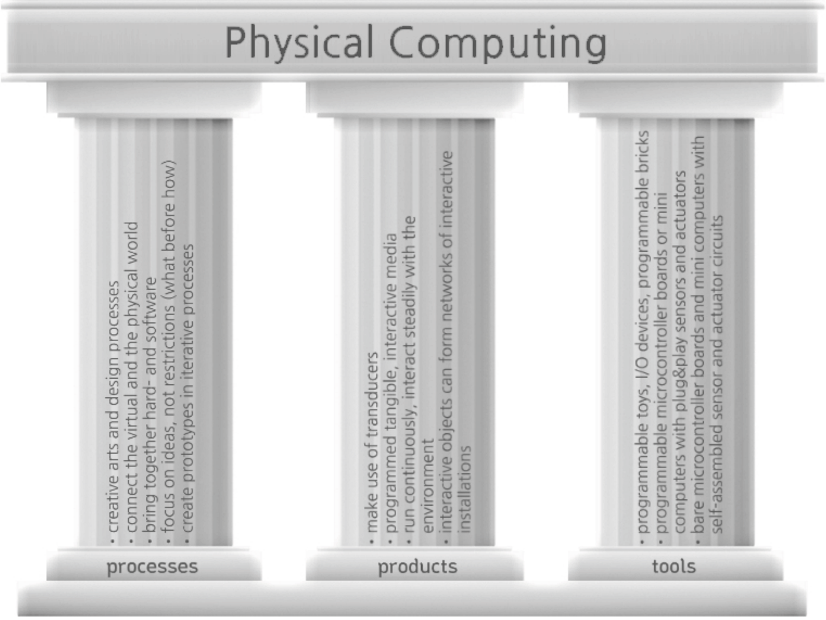

Physical Computing
• creative arts and design processes
• connect the virtual and the physical world
• bring together hard- and software
• focus on ideas, not restrictions (what before how)
• create prototypes in iterative processes
processes
• make use of transducers
• programmed tangible, interactive media
• run continuously, interact steadily with the environment
• interactive objects can form networks of interactive installations
products
• programmable toys, I/O devices, programmable bricks
• programmable microcontroller boards or mini computers with plug&play sensors and actuators
• bare microcontroller boards and mini computers with self-assembled sensor and actuator circuits
tools

Figure 3: Three pillars of physical computing

3 Key Competences in Physical Computing

In Germany, throughout the last years there have been discussions about the introduction or abolishment of computer science as a compulsory subject in public schools. These discussions are often about the general value of the subject and key competences of which learners benefit in all areas of life as opposed to expert knowledge, which is not relevant for most pupils unless they decide to study computer science. In order to highlight these competences in the thematic area of physical computing, the “Operational Definition of Computational Thinking for K-12 education” (International Society for Technology in Education (ISTE) & Computer Science Teachers Association (CSTA), 2011), the “Uniform Test Requirements for Informatics” (KMK - Ständige Konferenz der Kultusminister der Länder der Bundesrepublik Deutschland, 2004) (Standing Conference), the “Computer Science Education Standards” (Gesellschaft für Informatik (GI) e.V., 2008) as well as “A Model Curriculum for K-12 Computer Science” (Tucker et al., 2003) will serve as guidelines. In the following subsections, those areas are highlighted where physical computing makes an outstanding contribution.

3.1 Understanding Computing Systems

Interactive objects or installations made with physical computing are entire computing systems containing hard- and software components that students can assemble themselves and investigate further. Depending on the level of complexity they undergo in the particular setting, they can come all the way from an intuitive understanding (e.g. when controlling a programmable toy) to a deep understanding of interactive computing systems (e.g. when constructing an intelligent letterbox). Particularly aspects of hardware design help them to develop the abilities to identify and understand interactive systems in their every-day environments.

3.2 Formulating Problems

With physical computing, the basic ability to precisely formulate problems is formed and practiced as a first step in the process of designing and creating interactive objects. Students are required to unambiguously describe what is supposed to happen from an outside perspective, thus they focus on the problem formulation separately from thinking about possible ways of problem solving.

3.3 Organizing and Analyzing Data

In physical computing projects data can be collected automatically with self-made weather stations, a voting system to elect the next class representative or an automatic traffic recorder to count the number of cars passing outside school. This way, students learn with real-world data collected in their own environment by measurement objects they have designed and built themselves. They will find out about the coding and decoding of data and information while working with sensors that deliver data, which need to be interpreted and actuators that receive data, which has to be generated from information.

3.4 Algorithmic Thinking

Algorithmic thinking is also a crucial element of the physical computing process. Students at any level are required to precisely describe series of events, both serial and parallel. Physical computing in particular demands students to develop algorithms that allow their objects to run continuously and interact steadily with the environment.

3.5 Effectiveness and Efficiency

Key aspects of computational thinking include identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources. In physical computing projects, ineffective or ineffi-cient solutions are particularly evident. Interactive objects or installations should give immediate feedback, may include concurrent processes and should include intuitive interfaces. If they fail to meet the expectations, e.g. due to the choice of inappropriate sensors, excessive pauses or delayed responsiveness, this is immediately noticeable.

4 Conclusion and Discussion

Physical Computing is a complex activity that requires learners to be aware of hard- and software issues at the same time. On the introductory level (primary school), students may work with programmable toys or programmable bricks and drag & drop programming environments to learn the fundamentals of algorithmic thinking. Construction kits that have pre-assembled sensors and actuators to be plugged into a microcontroller either directly or with a shield allows older children to come to visible and tangible achievements very quickly. With advancing in physical computing both, on the hard- and the software side, students undergo learning processes that strengthen computational thinking and key competences that are necessary for all aspects of life. Physical computing can enrich future informatics classrooms with valuable competences that are focused in computer science education better than in many other subjects, as they are innate in the subject and not to be imposed artificially. Future research on this topic will therefore investigate how students of different age groups experience physical computing activities and what learning processes they undergo.

References

Banzi, M. (2011). *Getting Started with Arduino* (2nd Edition). Sebastopol, CA: O'Reilly Media/Make.

Computer Laboratory, University of Cambridge (2013). Physical Computing with Raspberry Pi. Retrieved October 24, 2013, from http://www.cl.cam.ac.uk/projects/raspberrypi/tutorials/robot/

Gesellschaft für Informatik (GI) e.V. (2008). Grundsätze und Standards für die Informatik in der Schule. Retrieved April 15, 2014, from http://www.sn.schule.de/~i-standard/docs/bildungsstandards_2008.pdf

International Society for Technology in Education (ISTE) & Computer Science Teachers Association (CSTA) (2011). Operational Definition of Computational Thinking. Retreived April 15, 2014, from http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf

KMK – Ständige Konferenz der Kultusminister der Länder der Bundesrepublik Deutschland (2004). Einheitliche Prüfungsanforderungen Informatik, 1–71. Retreived April 15, 2014, from http://www.kmk.org/fileadmin/veroeffentlichungen_beschluesse/1989/1989_12_01-EPA-Informatik.pdf

Libow Martinez, S., Stager, G. (2013). *Invent to Learn – Making, Tinkering, and Engineering in the Classroom. Torrance,* CA: Constructing Modern Knowledge Press.

O'Sullivan, D., Igoe, T. (2004). *Physical Computing: Sensing and Controlling the Physical World with Computers.* Boston: Thomson Course Technology PTR.

Papert, S., Harel, I. (1991). Situating Constructionism. In S. Papert, I. Harel (Eds.), Constructionism. Norwood: Ablex Publishing Corporation. Retrieved from http://web.media.mit.edu/~calla/web_comunidad/Reading-En/situating_constructionism.pdf

Przybylla, M., Romeike, R. (2012). My Interactive Garden – A Constructionist Approach to Creative Learning with Interactive Installations in Computing Education. In Kynigos, C., Clayson, J. E., Yiannoutsou, N. (Eds.), *Constructionism: Theory, Practice and Impact. Proceedings of Constructionism 2012* (pp. 395–404).

Resnick, M. (1996). Distributed Constructionism. In *ICLS '96 Proceedings of the 1996 international conference on Learning sciences* (pp. 280–284). International Society of the Learning Sciences.

Siemers, C. (2012). Handbuch Embedded Systems Engineering V 0.61a. Retrieved May 26, 2014, from http://www.in.tu-clausthal.de/uploads/media/Embedded_Systems_Engineering_Handbuch_V0_61a.pdf

Tucker, A., Deek, F., Jones, J., McCowan, D., Stephenson, C., Verno, A. (2003). A Model Curriculum for K–12 Computer Science: Final Report of the ACM K-12 Task Force Curriculum Committee. (Tucker, A. Ed.). New York: ACM. Retrieved from http://csta.acm.org/Curriculum/sub/CurrFiles/K-12ModelCurr2ndEd.pdf

Biographies

Mareen Przybylla is research associate and doctoral student at the professorship for Didactics of Computer Science at the University of Potsdam, Germany. Her main research interest is on physical computing in computer science education and its effects on students.

Ralf Romeike is the head of the Computing Education Research Group at the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Germany.

Copyright