# Protobject: A Sensing Tool for the Rapid Prototyping of UbiComp Systems

**Alessio Bellino**
DISCo, University of Milano-Bicocca
Viale Sarca 336/14, Milan, Italy
bellino@disco.unimib.it

## Abstract
The early stages of design of novel products in the fields of smart home and internet of things are usually carried out doing trials and errors. Accordingly, design tools should be flexible and quickly adaptable to any unexpected change of context or environment. With the aim to make the early design stages of prototyping easier and more flexible, we introduce Protobject, a rapid prototyping tool able to sense interactivity with existing stationary objects letting users reinvent their usage. Protobject works as a multipurpose camera-based sensor able to recognize any visually observable properties of stationary objects around us (e.g. white goods, lights, windows, curtains). In particular, Protobject can sense their state (e.g., lights on/off) and users' interactions with them (e.g. when turning on a light).


*Ubicomp/ISWC '16* Adjunct, September 12-16, 2016, Heidelberg, Germany
ACM 978-1-4503-4462-3/16/09.
http://dx.doi.org/10.1145/2968219.2971373

## Author Keywords
Rapid Prototyping; Sensing; IoT; Smart Home

## ACM Classification Keywords
H.5.2 User Interfaces: Prototyping

## Introduction
Interaction with the everyday environment is a great source of context. A large body of work has demonstrated how interaction with everyday objects can be leveraged to support users with context-aware applications (e.g. [4]). For example, when receiving a call, you pick up the phone. This means that you are busy answering the call. This information could be used to change the Skype state from "available" to "do not disturb" automatically. A wide range of works has explored how the design and development of context-aware behaviour in home environments can be supported (e.g. [1]). Moreover, the emergence of rapid prototyping tools (also commercial ones such as Arduino and 3D printers) made the development of these systems cheaper, easier and faster.

## Rapid Prototyping with Protobject
Protobject [3] is a rapid prototyping tool that enables observation of the state of any stationary object (e.g.

### Usage scenarios of Protobject

Several usage scenario were designed to validate Protobject use. Protobject allows users to make prototypes reinventing existing objects without using wires, sensors, or altering the environment. Protobject makes the early stages of Prototyping easier, quicker and more flexible.

**1. Smart umbrella stand**: The smart umbrella stand sends a notification to the user's smartphone if (1) he/she leaves home and (2) forgot the umbrella when (3) it is raining (weather.com API can be used for this scope).

**2. Tangible interface to control a light:** A tap and a pen are put in a corkboard and leveraged to control colour and intensity of a light displayed on a PC screen.

**3. Energy saving tool:** The tool makes a crosscheck to understand if a window is open while the heater is working signaling the waste.

white goods, doors, lights, windows, curtains, umbrella stands) as input for prototyping and testing smart systems. In many cases, Protobject can be used to sense also users' interactions with objects. In fact, some interactions can be detected according to changes of state: for example, closing a door (interaction), its state changed from "open" to "close"; taking an umbrella (interaction), its state (in the umbrella stand) changes from "present" to "absent". Protobject is designed to exploit state changes for prototyping scope, i.e. triggering events. For example, when John picks up the phone receiver answering a call (state change), Skype status changes to "Busy" automatically (triggered event).

We briefly summarize the main features of Protobject:

- It uses cameras as a generic sensor and, in particular, Wi-Fi cameras or smartphones. The latter can be easily turned into Wi-Fi camera by installing specific apps. Even old, smartphones can be used for the purposes of Protobject at no cost.
- Users are allowed at segmenting Protobject video stream in different regions in order to sense different observable properties at the same time using a single camera.
- Optionally, Protobject lets users add markers to objects to enable tracking of additional states that, otherwise, would be not observable in the object itself (e.g. rotation of a symmetrical object, etc.).
- The training of Protobject is carried out by demonstration of the different states of interests. At design time, Protobject learns the states of an object as visual images. At run time, Protobject matches the camera live stream against the stored image of states to detect, over time, which the current state is.
- Protobject allows the recognition of generic presences such as someone in front of a door, on a sofa, or a smartphone on a table.
- When necessary, Protobject can be used in addition to Arduino for extending its potentialities.
- Protobject frequency recognition can range from 30 milliseconds to one second being adapt for real-time usages.
- Protobject is able to sense objects without the need to alter them or the surrounding environment physical. Moreover, Protobject is flexible and easily adaptable to any change of environment.

Protobject uses deliberately a 'low tech' approach: smartphones, low skill requirements, existing objects to be reinvented and reused, and coloured markers that can be easily printed off, i.e., all resources widely available. Unlike other camera-based approaches for prototyping (e.g. [2]), Protobject aims to learn and detect states of objects easily and quickly.

## The User Interface

Protobject works in two modalities: design and detection mode. The design mode lets Protobject record the possible states of the objects (i.e., building databases of states). The detection mode lets Protobject detect states of objects according to the states previously recorded in the design mode.

*Design Mode*

After positioning the camera towards the target objects, users can create new regions (e.g. the semi-transparent regions on the plug and the lamp in Figure 1→1 and 2) drawing them with the mouse. Next, users can assign a name to the region just created (e.g. Lamp, Figure 1→3). For each region, users must record

the possible states they want to detect (by clicking on the button in Figure 1→4). Each state (e.g. "Off" in Figure 1→5 and "On" in Figure 1→6) is labelled with a value that can be edited in the fields next to them (Figure 1→7 and 8). Of course, interacting with the objects is needed to record these 'pictures of states'. For example, to build the states of the lamp in Figure 1, it is needed to turn it on to record the "On" state and turn it off to record the "Off" state. Once a new object is created and defined, it can be saved (see Protobject toolbar). Protobject saves the regions (position and

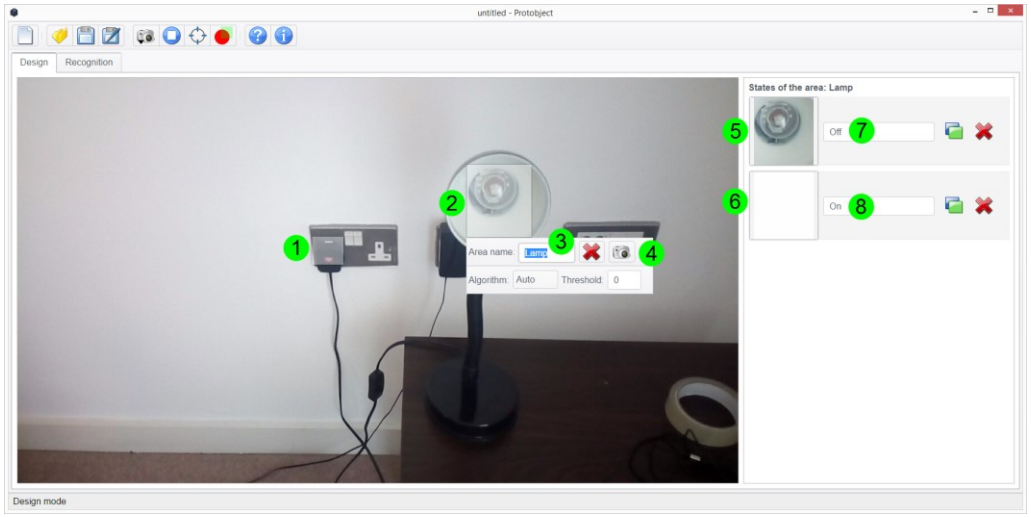


Figure 1. Protobject learns how to detect the state of a lamp and a plug.

dimension) with the recorded images of the states (database of states). Moreover, objects previously saved can be loaded, or new (blank) objects can be created.

*Recognition Mode*

Enabling the recognition mode, each region displays the current state (e.g. the state labels previously inserted in Figure 1→7 and 8) in real time (e.g. Figure 2→1 and 2). States are refreshed at a certain frequency (Figure 2→3). Enabling the WebSocket output (Figure 2→4), Protobject can send the detected states to other components (e.g. external applications that behave according to the received states).

## Usage Scenarios

According to the design of several usage scenarios, we can state that detecting and leveraging changes of states resulted to be simple and fruitful for prototyping IoT and Smart Home systems. The video (https://youtu.be/u2ra5L6rMrs) displays some of them, which are also summarized in the previous page.

## Protobject Design Approach

In general, while designing, makers and designers may wish to focus just on scenarios, interactions, and the operating way of interactive products rather than on electronics, sensors, or fabrication processes. In this regard, Protobject lets users rapidly change the environment settings without being concerned about sensors, wires and the general settings of the environment. Protobject also lets users reinvent and reuse existing objects without any (irreversible) modification.

## Reliability of the Recognition

When prototyping, environmental light conditions are normally stable and Protobject rarely makes recognition errors. At any rate, there could be situations in which light conditions have to change, also when prototyping. This is why we evaluated Protobject recognition

simulating different lighting variances to measure its robustness. In particular, we evaluated recognition errors during the detection of states.

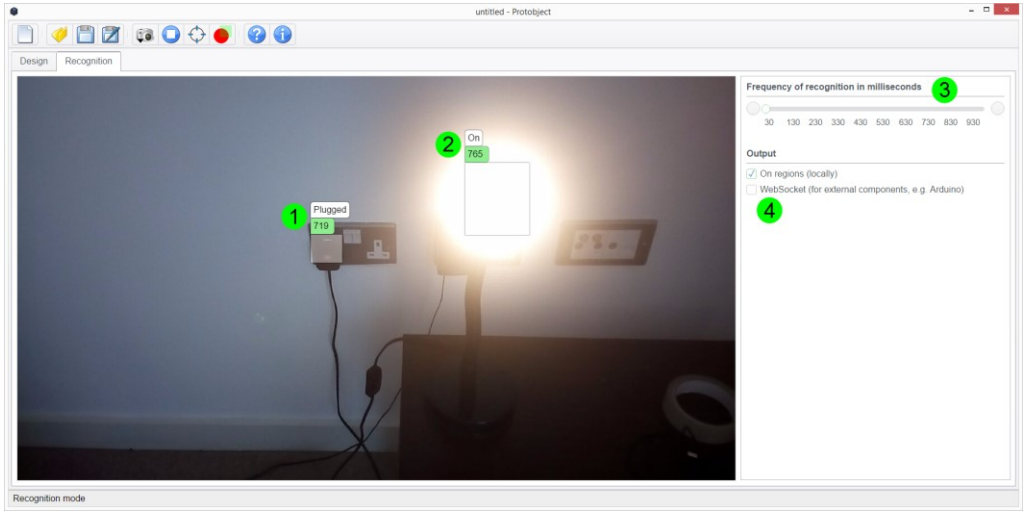


Figure 2. Protobject recognizes the state of a lamp and a plug.

*State Recognition Evaluation*

In order to have real prototyping conditions, we performed these evaluations in a domestic environment. We tested the recognition of the states of several objects and in particular, we detected the:

1. amount of water from a tap;
2. position of a curtain;
3. interaction with a mechanical button;
4. state of a LCD display;
5. function knob position of an oven;
6. colour of a surface;
7. knob position using a colour-wheel-based marker;
8. level of a coloured liquid;
9. opening angle of a door;
10. presence and colour of towels in a towel rack.

Tests were carried out using a THL 5000 smartphone as Wi-Fi camera. For each object, we recorded the database of states at the best possible light conditions in the environment. Next, after setting an object at a certain state, we tested whether that state was actually recognized correctly and stably (1) at the same lighting condition of learning, (2) while simulating shadows using a cardboard and (3) while a person was moving around the object.

The evaluations were promising. Accordingly, we can state that Protobject is able to ensure a reliability good enough for the early rapid prototyping of interactive products also in the case of lighting variances caused by the cardboard or people around.

## Acknowledgements

Thanks to Giorgio De Michelis, Flavio De Paoli and Hans Gellersen for some pieces of advices. Thanks to colleagues, friends and The FabLab of Milan for giving me many inspirations. Thanks to Daniela Bascuñan for reviewing my English.

## References

1. Dey, A.K. et al. "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications." H-CI 16.2 (2001).
2. Maynes-Aminzade, D. et al. "Eyepatch: prototyping camera-based interaction through examples." Proc. of the 20th annual ACM symposium on User interface software and technology. ACM, 2007.
3. Protobject Website, https://protobject.com
4. Weiser, M. "The computer for the 21st century." Scientific American 265.3 (1991).