

# Mobile Service Robot State Revealing Through Expressive Lights: Formalism, Design, and Evaluation

Kim Baraka<sup>1</sup>  · Manuela M. Veloso<sup>2</sup>

Accepted: 28 September 2017 / Published online: 16 October 2017  
© Springer Science+Business Media B.V. 2017

**Abstract** We consider mobile service robots that carry out tasks with, for, and around humans in their environments. Speech combined with on-screen display are common mechanisms for autonomous robots to communicate with humans, but such communication modalities may fail for mobile robots due to spatio-temporal limitations. To enable a better human understanding of the robot given its mobility and autonomous task performance, we introduce the use of lights to reveal the dynamic robot state. We contribute expressive lights as a primary modality for the robot to communicate to humans useful robot state information. Such lights are persistent, non-invasive, and visible at a distance, unlike other existing modalities. Current programmable light arrays provide a very large animation space, which we address by introducing a finite set of parametrized signal shapes while still maintaining the needed animation design flexibility. We present a formalism for light animation control and an architecture to map the representation of robot state to the parametrized light animation space. The mapping generalizes to multiple light strips and even other expression modalities. We demonstrate our approach on CoBot, a mobile multi-floor service robot, and evaluate its validity through several user studies. Our results show that carefully designed expressive lights on a mobile robot help humans better understand robot states and actions and can have a desirable impact on a collaborative human–robot behavior.

**Keywords** Robot transparency · Expressive lights · Non-verbal robot communication · Mobile service robots

## 1 Introduction

### 1.1 Motivation

Mobile robots are entering our daily lives and are expected to carry out tasks with, for, and around humans in environments such as hospitals, supermarkets, hotels, offices, and shops. For effective operation of these robots in these human-populated environments, it is important that humans have an understanding of some of the processes, information, or decisions taking place on the robot. Due to their mobility and possible diversity of states, as well as actions, while evolving in a dynamic environment, *revealing information about a robot's state* over the course of its task execution is crucial to enable: (1) effective collaboration between humans and the robot, (2) better trust in the robot, and (3) more engaging human–robot social interactions.

Central to the scope of this work is the idea of *expression*, which we think of as *externalizing hidden information of an agent*, in our case a mobile robot. We will be discussing *expressive behaviors*, in other words robot behaviors that have a specific communicative purpose about the robot itself, in particular its state and actions, as well as *expression channels* available to the agent to use those expressive behaviors. Current expression channels for mobile robots mainly include speech and on-screen display. However, when it comes to mobile robots traveling long distances, these expression channels may fail for different reasons. First, humans are not always in close proximity to the robot, in which case the speech might be inaudible and the on-screen text not visible. Second, speech is transient in that its associ-

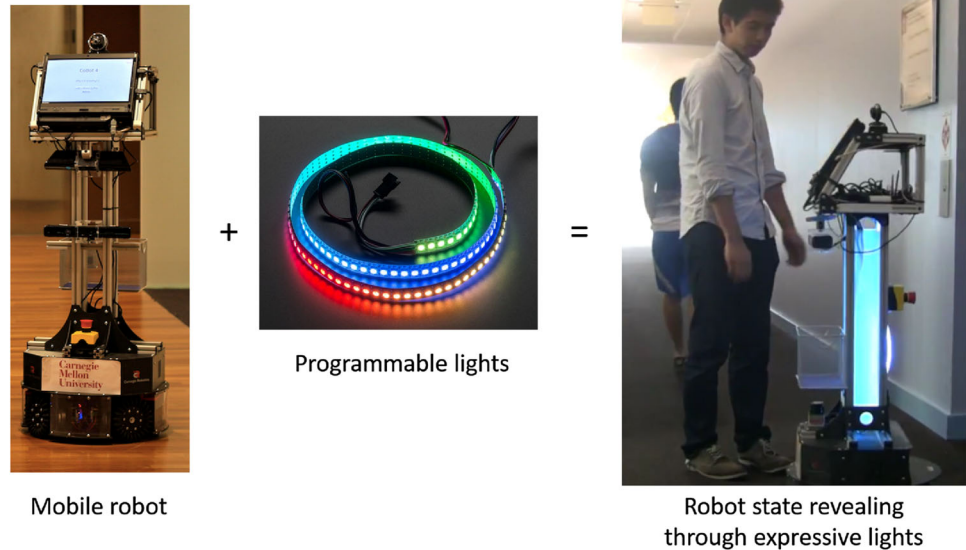
---

✉ Kim Baraka  
kbaraka@andrew.cmu.edu

<sup>1</sup> Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA

<sup>2</sup> Machine Learning Department, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA

**Fig. 1** Augmenting a mobile robot's communication capabilities with a new modality: expressive lights



ated expressive behaviors only last the duration of a sentence. The *mobile* aspect of these robots, also shared by other embodied agents such as self-driving cars or aerial robots, hence calls for other methods of communication that are both persistent and perceivable at a distance.

To remedy these problems, we contribute *expressive lights* as an expression channel communicating useful information about a mobile robot's state (see Fig. 1). Such lights are *visible at a distance*, provide a *persistent* visualization of the state information, and are *non-invasive* as compared to other possible modalities such as loud non-verbal expressive sounds. By using such lights, we are effectively enabling the robot to *modify its appearance* as a method of communication with humans, which is a distinguishing feature for expression as compared to other modalities. Literature has shown that abstract dynamic visual cues [23], and more specifically dynamic lighting [29], have been shown to elicit interactive social responses. These results potentially suggest that *expressive lights* on a robot are also likely to create more engaging interactions with humans if, first, they are communicating useful information - in other words they are *informative* - and, second, if they do so in a *legible* (i.e., readable) manner. These persistent lights may also serve as a complement to existing modalities of interaction that are transient (e.g., speech) or that require close proximity (e.g., on-screen text).

Throughout this paper, we will focus our analysis on an *autonomous mobile service robot*, CoBot, which provides different types of services to people in a building across multiple floors. However, the ideas we present are made general enough to be easily ported to different platforms and designs than the ones adopted in this paper. Nevertheless, CoBot constitutes an interesting platform for the purpose of expression of the state information to humans through lights. Indeed, the

ways in which CoBot interacts with humans are diverse: it can ask for help from humans when facing limitations (so-called *symbiotic autonomy* [27]), influence change in human behavior for the needs of its tasks, or provide useful information for the task at hand. The spectrum of information potentially communicated through expressive lights is hence greatly diverse and non-simplistic.

In summary, this paper will be investigating the use of lights to make elements of a robot's state visible to humans, in an abstracted fashion. We focus on light animations that are *informative*, i.e., providing useful information for the situation at hand, and *legible* i.e., whose meaning is intuitive and doesn't require prior training.

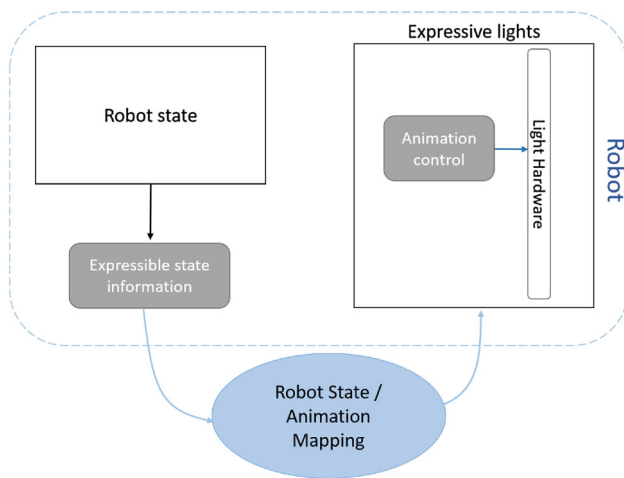
## 1.2 Approach

Figure 2 summarizes our approach to map the robot state information to expressive light animations. The relevant state information is first formalized in a format which makes it easy to map to an expressive behavior such as a light animation representing a concrete state of the robot. We then provide a formal control method for light strips, on which we base our mapping from robot state to light animation. The mapping and the designed expressive light animations are finally designed, validated, and tested, by running appropriate user studies.

## 1.3 Contributions

The contributions of this paper are as follows:

1. An *efficient robot state representation and mapping method* suited for expression of state the information,



**Fig. 2** Overview of our robot state/light animation mapping approach

2. A formal framework for animation control focusing on addressable light strips and generalizable to other forms of light arrays,
3. User studies to investigate the design and impact of animating robot state through expressive lights.

#### 1.4 Reader's Guide

The rest of the paper is organized as follows:

- Section 2 summarizes relevant literature on expressive lights, their use in technology and robots, as well as general non-verbal robot communication.
- Section 3 presents a general formalism for a mobile robot's state suitable for expression of the state information.
- Section 4 lays out our formalism for light animation and specifically control of addressable colored light strips.
- Section 5 presents our theoretical framework for mapping robot state to an expressive behavior, such as a light animation.
- Section 6 reports three studies that were conducted to inform: (1) the design of appropriate light animations (parameter selection), (2) their evaluation and generalization to similar scenarios, and (3) their impact on human behavior in the real world.
- Section 7 reports three studies that we conducted to inform: (1) the design of appropriate light animations (parameter selection), (2) their evaluation and generalization to similar scenarios, and (3) their impact on human behavior in the real world.

## 2 Related Work

This paper will be focusing on the use of lights as a communication medium for mobile robots. In this section, we therefore first present a short overview of general uses of lights in different applications, and how they have previously been used as an expressive medium. We then focus our review on lights on robots and other existing expressive non-verbal modalities.

### 2.1 Short Survey of Uses of Lights

In this subsection, we discuss different general uses of lights in technology.

#### 2.1.1 Lights for Communication at a Distance

Light signals have been widely used in the history of mankind to convey information at a distance or in low visibility environments, such as in aviation and maritime navigation [21], where the use of acoustic signals is not possible because of signal attenuation. However, most of these signals often need to be learned since they rely on codes (an extreme case being Morse code communication before radio technologies existed).

Nowadays, we see lights in our daily lives for communication at a distance, especially on roads, thanks to indicators such as traffic lights that control traffic flow through a simple color code, or lights on cars such as flashers communicating upcoming driver actions, emergency flashers expressing an unusual or dangerous situation, brake warning rear lights indicating the status of the brake, and headlights sometimes used by drivers to acknowledge or communicate with other cars or pedestrians.

Light also plays a role in some biological systems, especially in animal communication. Through a process called bioluminescence [22], some animal species such as jellyfish, octopus, anglerfish, or fireflies can emit light to deceive, attract, or communicate different messages to other animals [13].

#### 2.1.2 Lights for Revealing State Information

As discussed in the previous paragraph, cars are a good example where lights are used to reveal information about the car or the driver's state. Similarly, personal electronic devices and appliances often make use of light indicators with usually intuitive, walk-up-and-use patterns to convey information to the user. We see light indicators on all sorts of devices including cell phones, washing machines, toasters, laptops, cameras, battery chargers and more.

Such uses pose the problem of a *mapping* from a concrete state of the device to an abstract visualization into a light

animation, which we will be investigating in Sect. 5. We can exploit the fact that humans generalize from their daily experience, to get inspiration in our light animation design from standard patterns, codes and meanings associated with some of these existing light behaviors.

### 2.1.3 Lights and Aesthetics

Because of their visual aesthetic appeal and wide flexibility in configuration and modularity, expressive lights have been featured extensively in contemporary art including interactive art installations [17], bridge art (e.g., the Pausch bridge at Carnegie Mellon University<sup>1</sup>) or large scale visualizations of data such as the state of the Internet [15]. Expressive lights have also been used on wearable apparel [7].

Stage and scene lighting share common expressive features with indicator lights like color, intensity and time-varying patterns [9], but there the purpose is to illuminate rather to use the light source itself as an expressive communication modality. The same holds for modern programmable luminaires, mainly used for creating diverse moods in physical spaces [16].

## 2.2 Light as an Expressive Medium

Because of the way humans process visual information, which includes color and motion as “undoubtable attributes” guiding the human attention process [34], programmable multi-color lights, which combine color and motion to create light animations, are a good candidate for changing a robot’s appearance in order to communicate different types of information to humans.

### 2.2.1 Light Control

Addressable LED technology has unlocked fine control of multiple light sources with possible color changes and gave rise to several control protocols of which DMX512 is probably the most popular, especially for large installations or multiple light devices.<sup>2</sup> Color Kinetics<sup>3</sup> offers a wide variety of tools to design and control large-scale light installations through various animations.

Smaller scale options include addressable RGB LED strips, on which we will be focusing in this paper. Each LED has a built-in microcontroller that controls the associated LED intensity as a function of time. By propagating serial communication packets throughout the strip, one can

achieve appealing light animations using only one physical data communication pin.<sup>4</sup>

### 2.2.2 Light Expression Space

Light signals generally allow for a large expression space with several degrees of freedom. For example, light communication in insects have been found to use modulations in spectral composition, brightness, shape, size, and timing [22]. Technologically speaking, there has been efforts to standardize color spaces (RGB, HSV etc.), but not much work has been done when it comes to standardizing light patterns [14]. Also, parametrized abstract motifs have been used for spatial layout modulation [5].

Current LED technology offers control of brightness and color for individual light sources and allows for customizable layout options,<sup>5</sup> unlocking very flexible designs for light animations.

### 2.2.3 Light Animation Semantics

Because of the wide use of lights to convey concrete information, it seems that humans tend to associate specific *meanings* to different light animations.

For a single light source of a fixed color, different light patterns seem to convey diverse information about a personal device’s operation [14]. In a traffic setting, flashing lights, for instance, seem to be associated with the idea of warning (such as an out-of-service traffic light or an emergency car flasher). Also, some studies have been conducted on the conspicuity of light patterns as a function of frequency, duration and contrast [12], but also on the perception of emergency warning signals, especially in terms of color combinations [8]. Color has also been associated to the expression of emotions in different contexts, such as general perception of color [36], uses in clothing [7], or on virtual agents [9,24].

Color theory [36], as well as learned color codes (e.g., traffic lights), provide a good starting point for the design of colored light animations carrying meaning (in our case related to robot state). However, it remains difficult to predict the appropriateness of colored animations for light sources extending in space beyond a single point (namely a light strip) and expressing meaning in relation to a complex machine such as a robot.

## 2.3 Robot Expression

In this subsection, we discuss different ways of expressing hidden robot information to humans.

<sup>1</sup> <http://www.cmu.edu/randyslecture/bridge.html>.

<sup>2</sup> <http://opendmx.net/index.php/DMX512-A>.

<sup>3</sup> <http://www.colorkinetics.com/>.

<sup>4</sup> <https://learn.adafruit.com/adafruit-neopixel-uberguide/advanced-coding>.

<sup>5</sup> <http://www.colorkinetics.com/>.

### 2.3.1 Lights on Robots

The use of lights for non-verbal communication on robots remains rudimentary. Most of these uses do not have a direct functional role but rather focus on creating abstract impressions (such as “artificial subtle expressions” [20]), expressing emotions [18], or serving as very basic indicators (such as for battery level). Often, we see these light expressions dissociated from the robot’s state, such as for instance expressing people’s emotions in a cafe-style room on a Roomba robot [26]. To the best of our knowledge, the only instances of functional and state-related light communication in robots are for human–robot speech synchronization using a Nao robot [11], and for communicating intent in robot navigation using a drone [32]. In [11], an animated LED is used to avoid utterance collisions in verbal human–robot communication by subtly blinking between the user’s speech end and the robot’s speech start. In [32], an array of LED’s are used to communicate direction of navigation on a quadcopter. This last work fits within our idea of expressing a part of the robot’s state through light animations. However, in the prior work, the expressed feature (directionality) remains a low-level one and the light expression has a low level of abstraction. In contrast, we will be focusing on higher-level features of the robot’s state related to the robot’s tasks in the (often unpredictable) environment.

### 2.3.2 Other Non-verbal Modalities for Robot Expression

Several non-verbal modalities have been considered for human–robot communication. Some of them, such as eye gaze [1], proxemics [6] are more suited for expressing an emotional or affective state of the robot. In this paper, we are interested in expressing robot states that are related to tasks, and as such, there exist two main other non-verbal modalities which could be used for such expression. The first modality is expressive motion, which has been studied in different settings such as manipulators expressing goal and planning information [10], as well as mobile robots expressing affective states [19]. The type of mapping problem in which we are interested in this paper has been studied for a continuous input signal such as music or speech being mapped to motion sequences on a humanoid robot [37]. The second modality is expressive sound and vibration [30], which have been less explored for robotics applications and are usually more prevalent in electronic devices (e.g., phone notifications).

## 3 Mobile Service Robot State and its Expressible Elements

In this section, we first introduce a representation of robot state that is suited for expression of elements of robot state

on a given expression channel (expressive lights in our case). Our analysis focuses on CoBot, a collaborative mobile service robot with diverse capabilities. We then discuss the process of mapping the robot state information to light animations with respect to the formalism presented in the previous section. We illustrate this mapping through the selection of *informative* elements of CoBot’s state to be displayed using expressive lights. We then discuss the generalizability of our mapping framework to multiple light strips/modalities expressing different aspects of the robot’s state. We end the section by providing details about how the mapping framework was implemented on the CoBot robot with two light strips expressing task-related and navigation-related aspects of the robot’s state, respectively.

### 3.1 CoBot Overview

In this paper, we use CoBot, a collaborative mobile service robot, as an example to illustrate the ideas presented. CoBot can perform a set of services to humans in a building across multiple floors. Building locations (rooms, kitchens, and elevators), as well the navigation map of the building, are known to the robot. CoBot robustly navigates autonomously [33] from location to location while avoiding obstacles during its navigation, whenever possible, or stopping in front of unavoidable obstacles such as humans obstructing its path. When facing limitations, the robot asks for help from humans. For example, unlike some other robots that can operate elevators [31], or load/unload objects on/from the robot, CoBot is unable to perform these tasks alone. However, it is able to overcome those limitations by proactively asking help. This is the main idea behind *symbiotic autonomy* [27], illustrated in Fig. 3.

#### 3.1.1 CoBot Tasks and Services

The *tasks* performed by CoBot are of one of two types:

- *Navigation tasks* involve navigating from a start location to a goal location according to a predefined navigation map on which a path planning algorithm is run, and
- *Human interaction tasks* involve asking for human help (referred to as an ‘ask’ task) or waiting for human input, such as confirmation, service initiation, or dismissal (referred to as a ‘wait’ task).

Tasks can be combined to form *services*. The three services offered by CoBot and considered in this paper are the following:

- *Go-to-Room* service, in which the robot goes from its current position to a goal location.



**Fig. 3** An example of symbiotic autonomy: CoBot getting help from a human at an elevator

- *Item-transport* service, in which the robot transports an item in its basket from a start location to a goal location.
- *Escort* service, in which the robot escorts a person from a start location (typically an elevator) to a goal location.

Figure 4 shows CoBot performing some of these services. An example of a service broken down into individual tasks is shown below.

*Example 1* Sample service: Transport an object from room 7002 (on the 7th floor) to 3201 (on the 3rd floor):

- Task 1: *Navigate* from current location to service start location (room 7002).
- Task 2: *Wait* for human to put the object to be transported in basket, and get confirmation.
- Task 3: *Navigate* from room 7002 to the 7th floor elevator.
- Task 4: *Ask* for human assistance to take the elevator to the 3rd floor.
- Task 5: *Navigate* inside the elevator.
- Task 6: *Ask* for human assistance to know when correct floor is reached.
- Task 7: *Navigate* out of the elevator.
- Task 8: *Navigate* from the 3rd floor elevator to service goal location (room 3201).
- Task 9: *Wait* for human to collect object and get service completion confirmation.

### 3.1.2 CoBot User Modalities

Robot services can be requested in three different ways, corresponding to three different input modalities:

- Through the robot’s touch screen: A Graphical User Interface (GUI) enables scheduling services from CoBot itself by letting users choose the service type and its required parameters. It can also be used to interrupt the execution of task if needed.
- Through the speech interface [25]: The GUI also includes a button that enables speech interaction with the robot. The user can issue service commands using simple structured language that the robot understands.
- Through a web interface: People in the building can schedule a robot service in a time window through a web interface.

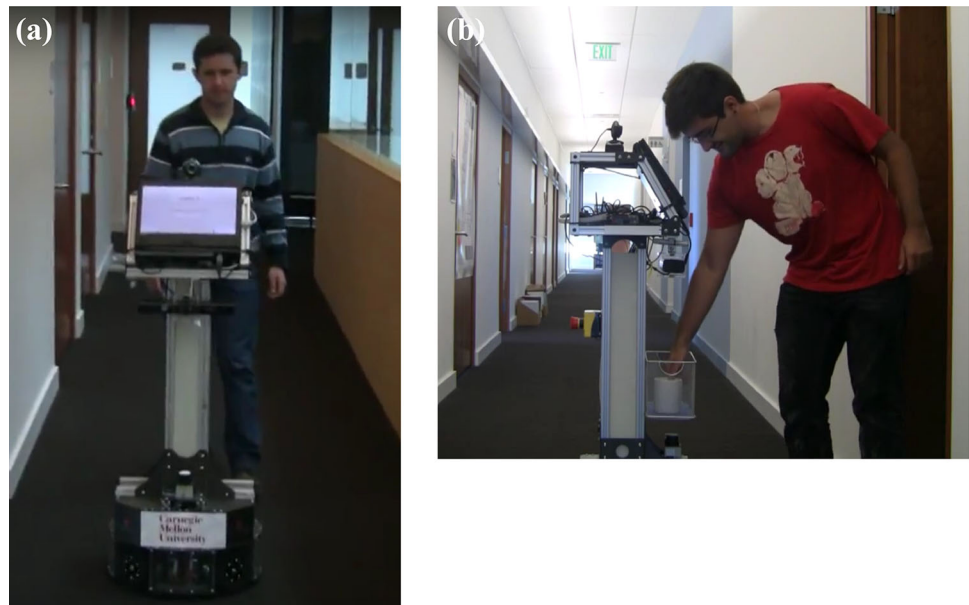
### 3.1.3 Robot Motion Modes

Mobility constitutes an important distinguishing feature of CoBot compared to other types of robots. We can distinguish three different motion modes in which the robot can be, summarized below.

- *Moving* in this mode, the robot is executing a navigation task successfully. We distinguish these two cases:
  - the robot moves *with someone* (escort service),
  - the robot moves *on its own* (all other tasks).
- *Stopped* in this mode, the robot is not moving, which can be due to several reasons. We also distinguish two cases:
  - the robot is *intentionally* stopped, either because it is idle, or because it is performing an interaction task.
  - the robot is *forced* to be stopped, because of some unexpected event such as the presence of an obstacle or internal failure. In the presence of an obstacle, the robot says “Please excuse me” to incite any human obstacles to move away. If it is blocked for more than a few minutes, it will send an email to the developers for help.

In the next section we present a formalism for CoBot’s state. The formalism presented is however made general enough to easily apply to other robots with different types of services and capabilities.

**Fig. 4** CoBot performing different services: (a) an escort service; (b) a transport service



### 3.2 Robot State Representation

#### 3.2.1 Robot Variables

**Definition 1** The *robot variables* represent any relevant quantity (discrete or continuous) that the robot maintains through its software. We categorize robot variables into *service variables* (related to the robot's services and tasks and the planning associated with them), *execution variables* (related to task execution in the environment), and *internal variables* (related to the internals of the robot's software and hardware).

The robot variables that we consider for CoBot are the following:

- *Service variables*
  - the current service type `service` ('go-to-room', 'transport', or 'escort'),
  - the current task type `task` ('navigate', 'ask', or 'wait'),
  - the service path plan `path-plan` (list of position vertices ( $x$ ,  $y$ , floor#)),
  - the service start location `start-loc`, and
  - the service goal location `goal-loc`.
- *Execution variables*
  - the current robot location `loc`,
  - the current robot speed `speed`
  - a boolean indicator `path-blocked` for whether the robot's path is blocked, causing the robot to stop,
  - the duration of the path blockage `block-time` (None if `path-blocked` is false),

- a boolean indicator `GUI-interrupt` for whether the robot's task was interrupted from the GUI, and
- the duration of the interruption from the GUI `interrupt-time` (None if `GUI-interrupt` is false).

- *Internal variables*

- the list `sens-status` of sensor statuses (0 for normal / 1 for faulty),
- the list `act-status` of actuator statuses (same),
- the software status `soft-status` (0 for normal / 1 for important error),
- The gravity of the fault if it occurs `fault-level` (e.g., on a scale from 1 to 5).
- a boolean indicator `charging` for whether the robot is charging, and
- the percentage battery level `batt-level`

Note that the value of some of these variables might be undefined depending on the situation the robot is in; in that case, we assign a value of None to those variables with undefined values.

#### 3.2.2 State Features and Robot State

We build upon the robot variables defined in the previous section to generate states in which the robot may find itself. The robot state is determined using state features, as defined below.

**Definition 2** *Robot state features* are discrete (usually high-level) aspects of the robot's state. They are represented as *logical expressions over robot variables*. The state features

we consider are only those who are relevant to humans that potentially interact with the robot such as users, humans in the navigation environment, and developers.

The state features for CoBot considered in this paper are listed below:

- ‘Escorting’:  $(\text{service} = \text{'escort'}) \wedge (\text{task} = \text{'navigate'})$ .

The robot is in the process of escorting the user.

- ‘Blocked by an obstacle’:  $(\text{path-blocked} = \text{True}) \wedge (\text{task} = \text{navigate})$ . An obstacle is impeding the navigation task progress.

- ‘Asking for help at an elevator’:  $(\text{task} = \text{'ask'}) \wedge (\text{isElevator}(\text{loc}))$  (where  $\text{isElevator}(a)$  returns True if location  $a$  is an elevator and False otherwise).

The robot is waiting to get help at an elevator.

- ‘Charging’:  $(\text{charging} = \text{True})$ .  
The robot is connected to power and charging its battery.
- ‘Interrupted by user’:  $\neg (\text{task} = \text{None}) \wedge (\text{GUI-interrupt} = \text{True})$ .

The robot has been interrupted by a user through the GUI.

- ‘Waiting for object loading’:  $(\text{service} = \text{'transport'}) \wedge (\text{task} = \text{'ask'}) \wedge (\text{loc} = \text{start-loc})$ .

The robot is waiting for the object to be transported at the service start location.

- ‘Stopped because of faulty component’:  $(\text{contains1}(\text{sens-status})) \vee (\text{contains1}(\text{act-status})) \vee (\text{contains1}(\text{soft-status}))$  (where  $\text{contains1}(a)$  returns True if list  $a$  contains at least one 1 and returns False otherwise).

One or more execution-time fault(s) occurred in the software or hardware of the robot (e.g., the LiDAR sensor gets reading errors when the robot navigates in areas with a lot of sunlight).

- ‘Waiting for dismissal’:  $(\text{task} = \text{'ask'}) \wedge (\text{loc} = \text{end-loc})$ .

The robot is waiting for the user to dismiss it by pressing its “Done” button on its touch screen.

- ‘Navigating’:  $(\text{task} = \text{navigate})$ .  
The robot is performing a navigation task.

- ‘Turning’:  $(\text{distanceFromNextVertex}(\text{loc}, \text{path-plan}) < d_{th}) \wedge (|\text{nextTurnAngle}(\text{loc}, \text{path-plan})| > \alpha_{th})$  (where  $d_{th}$  and  $\alpha_{th}$  are thresholds for the distance to the next vertex in  $\text{path-plan}$  and for the upcoming turning angle;  $\text{distanceFromNextVertex}(\cdot, \cdot)$  returns the distance between the current location  $\text{loc}$  and the next vertex in the  $\text{path-plan}$  and  $\text{nextTurnAngle}(\cdot, \cdot)$  returns the upcoming turn angle  $\alpha$  shown in Fig. 5).

The robot is about to take a turn in its navigation path.

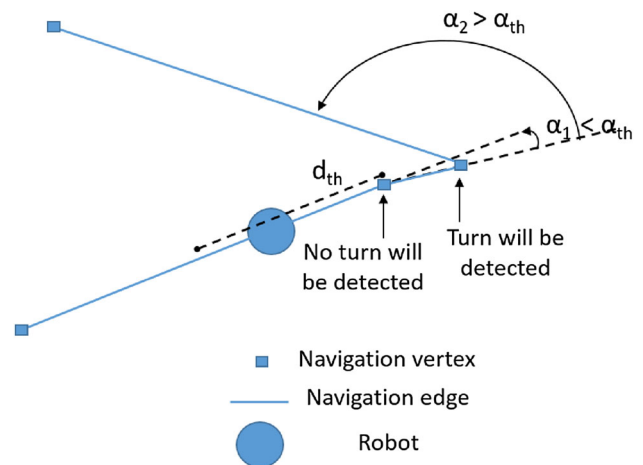


Fig. 5 Turn detection check used for computation of feature ‘Turning’

Note that state features are not necessarily mutually exclusive: more than one state feature could be true at the same time. Since this might pose a problem for the purpose of expression (we can only visualize one or at most a few of these features at a time), we will handle this issue in Sect. 5.

**Definition 3** The robot *state* is defined as the set of all state features that are true at a given time  $t$ . Since the state of the robot is constantly changing as a function of time, the state is a dynamic process  $S(t) = \{s_1, s_2, \dots, s_{m_t}\}$  where  $s_{1..m_t}$  are the state features true at time  $t$ .

### 3.3 What Part of the Robot State to Express?

So far, in this section, we have looked at robot state dissociated from robot expression. Even though state features have been defined to be relevant to users (and hence would gain at being expressed to them in some way), the actual nature of the medium used for expression hasn’t been taken into account yet in our analysis.

**Definition 4** We denote by *expression channel* any communication medium (verbal or non-verbal) that can be potentially used to express elements of a robot’s state. The expression channel we consider is *expressive lights*.

In particular, in this section, we are interested in two aspects of state representation for the purpose of expression:

- *Modulation* State features are a discrete representations of high-level, user-relevant elements of the robot’s state. However, this representation is rigid and doesn’t allow for modulation (no possible expression of “microstates” within the state feature or of continuous quantities that might be relevant when a particular state feature is true). For this reason, in this section we will be introducing



what we call “expressible state tuples”, a data structure which takes into account modulation of expressive behaviors (such as light animations) according to modulation quantities.

- *Simplification* For a given expression channel, there is a trade-off between complexity of the expression vocabulary (total number of different expressive behaviors used) and legibility of expressive behaviors (how readable or intuitive these expressive behaviors are). For better legibility, it might be useful to group or cluster some expressible elements of states together into classes which are expressed using the same expressive behaviors. For this reason, in this section we will be introducing what we call “expressible classes”.

### 3.3.1 Expressible State Tuples

We introduce *expressible state tuples*, which are tuples containing all state information relevant to a particular robot situation and expressible on a given expression channel.

**Definition 5** An *expressible state tuple* on a given expression channel is defined as a tuple  $\langle s, \mathbf{v}_s \rangle$ , where  $s$  is a state feature and  $\mathbf{v}_s$  is a vector of *modulating variables*, relevant for expression of state feature  $s$  on the considered expression channel. These additional variables can either be robot variables (defined in Sect. 3.2.1) or variables computed from robot variables, referred to as computed variables.

To illustrate the concept of an expressible state tuple, here are a few examples based on the state features listed in Sect. 3.2.2:

- $\langle \text{'Escorting'}, (\text{percentDone}(\text{loc}, \text{path-plan})) \rangle$ , where  $\text{percentDone}(\cdot, \cdot)$  computes the percent distance traveled along the path plan (progress so far towards goal location); this expressible state tuple could be translated into some sort of progress indicator visible to the escorted user.
- $\langle \text{'Blocked by an obstacle'}, (\text{block-time}) \rangle$ ; this expressible state tuple could be translated into a blockage indicator which gets more noticeable as the blockage time increases.
- $\langle \text{'Charging'}, (\text{batt-level}) \rangle$ ; this expressible state tuple could be translated into a visual indicator changing as the percentage battery level increases.
- $\langle \text{'Turning'}, (\text{nextTurn}(\text{loc}, \text{path-plan}), \text{speed}) \rangle$  (where  $\text{next-turn}(\cdot, \cdot)$  is a function returning 'left' or 'right' according to the direction of the upcoming navigation turn); this expressible state tuple can be translated into a turn indicator showing the upcoming turn direction.

### 3.3.2 Clustering Expressible State Tuples: Expressible Classes

For decreased complexity and increased legibility of expressive behaviors, we introduce in this section classes of expressible state tuples, or *expressible classes* for short. The clustering is dependent on the expression channel considered (different expression channels might require different levels of abstraction depending on their complexity/legibility tradeoff). We cluster in the same class those expressible state tuples which possess semantic similarities, allowing us to express them in a similar fashion through the expression channel. The expressible classes suggested below were designed for expressive lights as our expression channel; for other expression channels, as noted above, the classification may vary, even for the same expressible state tuples.

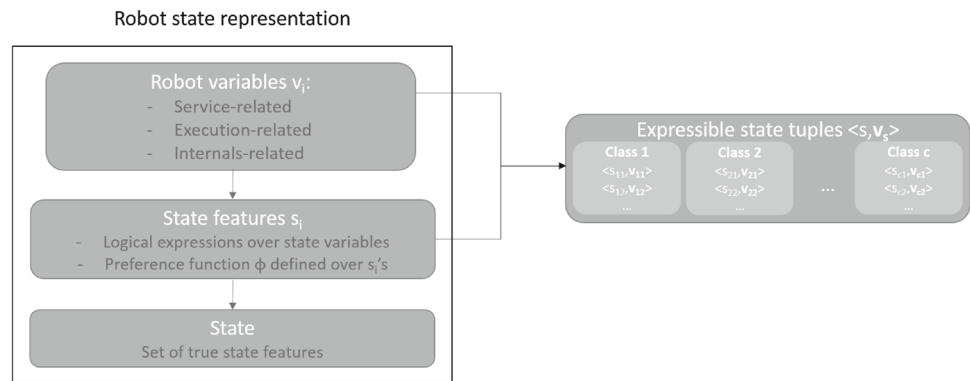
Based on the expressible state tuples listed in the previous subsection, we propose the following expressible classes:

- Class **'Progressing through a process with known goal'**: There are different kinds of processes that the robot goes through in which the goal is known. The progress on such processes could be expressed in the same way across different kinds of processes such as when the robot is escorting a person to a specific location or when it is charging its battery towards the maximum charge level. For both of the escorting and charging cases, the additional variable in the expressible state tuple represents a percentage completion. The expressible state tuples corresponding to this expressible class are:
 

```
{ 'Escorting', (percentDone(loc, path-plan)) }
{ 'Charging', (batt-level) }
```
- Class **'Interrupted during task execution'**: There are different ways in which the robot's task execution gets interrupted. From the perspective of expressive lights, it doesn't matter knowing why the interruption occurred (e.g., because of an obstacle, a faulty robot component, or a user-initiated interruption through the GUI) as much as communicating a state of interruption. (Other expression channels such as screen display or voice could eventually complement the expressive behavior with additional information.) The expressible state tuples corresponding to this expressible class are:
 

```
{ 'Blocked by an obstacle', (block-time) }
{ 'Interrupted by user', (interrupt-time) }
{ 'Stopped because of faulty component', (fault-level) }
```

**Fig. 6** Summary of the introduced robot state concepts



- Class ‘**Waiting for human input**’: As described previously, there are several situations for which the robot is waiting for some sort of human input (corresponding to tasks of type ‘wait’ or ‘ask’): the robot can be waiting for a task to be initiated by a user or confirmed at completion time, or it can be waiting for help in cases where it uses symbiotic autonomy to overcome some of its limitations. The expressible state tuples corresponding to this expressible class are:

```
{ 'Asking for help at an elevator',
  None }
{ 'Waiting for object loading', None }
{ 'Waiting for dismissal', None }
```

### 3.4 Section Summary

In this section, we focused on extracting *informative* elements of a robot’s state to be expressed through an expression channel, such as expressive lights. We presented a general representation of a mobile service robot’s state (robot variables, state features, expressible tuples) that is suited for expression on an expression channel (Fig. 6).

## 4 Animating Light Sources

The goal of this section is to provide a framework for animating a set of fixed light sources. This framework will help us design appropriate animations in a simple fashion and facilitate the process of mapping robot state space to light animation space. We begin by defining the concept of a light animation and reduce the dimensionality of the very large animation space by introducing a finite number of parametrized signal shapes. We then present the algorithms used to dynamically control a digital light strip according to the presented animation options. Our framework can be easily extended to other signal shapes or to multiple light strips.

## 4.1 Light Animation and Animation Space Definitions

### 4.1.1 Light Animation as a Continuous Intensity Function Matrix

**Definition 6** An animation  $A(t)$ , within a three-dimensional color space, of a set of  $n$  fixed point source lights is defined as a time-varying  $n \times 3$  matrix of light intensities:

$$A(t) = \begin{pmatrix} i_{1c_1}(t) & i_{1c_2}(t) & i_{1c_3}(t) \\ i_{2c_1}(t) & i_{2c_2}(t) & i_{2c_3}(t) \\ \vdots & \vdots & \vdots \\ i_{nc_1}(t) & i_{nc_2}(t) & i_{nc_3}(t) \end{pmatrix} \quad (1)$$

where the rows represent the indices of the individual point source lights or *pixels* and the columns represent dimensions of the color space or *color channels*  $c_1$ ,  $c_2$  and  $c_3$  (e.g., RGB, XYZ [28]). The intensity values represent a percentage of an allowed maximum intensity, and hence:

$$\forall t: 0 \leq i_{jc_k}(t) \leq 100 \quad j = 1, \dots, n; \quad k = 1, 2, 3$$

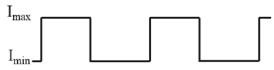
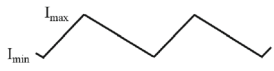



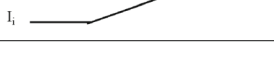
### 4.1.2 Spatial Layout

The animation matrix above does not capture the spatial layout of these pixels, which could be arranged in a linear, planar or three-dimensional fashion. For the rest of this work, we will focus on linear light strips. They simplify the analysis and representation of light animations and allow for greater mounting flexibility from a physical point of view. For linear strips, we let the pixel index (row index of the animation matrix) represent the position of the pixel on the strip, along a predefined direction.

### 4.1.3 Animation Space Intensity Functions

The space to which  $A(t)$  belongs is very large, as each of the intensity functions  $i_{jc_k}(t)$  can be any continuous function of  $t$  bounded between 0 and 100 and hence belongs to an infinite-dimensional space. As a result, we will only focus

**Table 1** List of parameterized signal shapes considered for each  $i_{j_{ck}}(t)$

Periodic	Blink (rectangle waveform) 
	Fade in/out (triangle waveform [possibly asymmetric]) 
	Smooth fade in/out (sinusoidal waveform) 
	Irregular Blink (modulated waveform) 
Non-periodic	Abrupt intensity/color change (step function) 
	Slow intensity/color change (clipped ramp function) 

on a limited set of possible intensity functions  $i_{j_{ck}}(t)$ , which we group into classes. We call these classes *parameterized signal shapes*, summarized in Table 1. There are plenty of light animations (either in libraries for light strips,<sup>6</sup> or in art projects using lights [15]) whose goal is to create aesthetically pleasing animations. These types of animations are usually sophisticated, fast-changing in space, time or color. We will be focusing on the simple signal shapes shown in Table 1 which, through efficient parametrization, will still provide us with great flexibility.

As a result of considering a handful of parameterized signal shapes, we have now reduced the dimensionality of the animation space drastically, and the analysis of this space is now simpler to work with.

#### 4.1.4 Animation Tuple Representation

For compactness of representation, we define an *animation tuple*, which fully characterizes the animation behavior for a subset of the light pixels.

**Definition 7** An *animation tuple* is defined as a tuple  $\langle sh, \mathbf{p}_{sh}, j_{start}, j_{end} \rangle$ , where:

- $sh$  is a string identifier for the signal shape used in the animation. It can take on the following values: “rect” for rectangle waveform, “tri” for triangle waveform, “sinl” for sinusoidal waveform, “modw” for modulated wave-

form, “step” for step function, and “ramp” for clipped ramp function

- $\mathbf{p}_{sh} = (p_1^{sh}, p_2^{sh}, \dots, p_{m_{sh}}^{sh})$  is a vector of parameters associated with that particular signal shape, where  $m_{sh}$  is the number of parameters for shape  $sh$
- $j_{start}$  and  $j_{end}$  are the indices of the start and end pixel, respectively, to be animated. In other words, the animation described by  $sh$  and  $\mathbf{p}_{sh}$  is applied to pixels  $j_{start}$  up to  $j_{end}$ , inclusive

The behavior of a full light strip is represented by a set of animation tuples  $\{ \langle sh_1, \mathbf{p}_{sh_1}, j_{start_1}, j_{end_1} \rangle, \dots, \langle sh_m, \mathbf{p}_{sh_m}, j_{start_m}, j_{end_m} \rangle \}$  such that:

$$\bigcup_{i=1}^m (\{j_{start_i}, \dots, j_{end_i}\}) = \{1, \dots, n\} \tag{2}$$

$$\bigcap_{i=1}^m (\{j_{start_i}, \dots, j_{end_i}\}) = \emptyset$$

where  $n$  is the number of pixels on the strip. In other words, first, the pixel indices must cover the whole strip and, second, there should be no overlap in the specified pixel ranges. We call an animation tuple set satisfying conditions 2 a *complete animation tuple set*. It fully characterizes an animation of a whole light strip.

## 4.2 Signal Shape Parametrization

We now present a detailed description of each of the parameterized signal shapes shown in Table 1. For periodic signals of period  $T$ , we refer to the portion of the signal extending from  $t = zT$  to  $t = (z + 1)T$  (where  $z \in \mathbb{Z}$ ) as a *cycle*.

### 4.2.1 Rectangle Waveform

The parameter vector  $\mathbf{p}_{rect}$  for this periodic waveform is composed of the following components:

- $(I_{c_1, min}, I_{c_2, min}, I_{c_3, min}) \equiv p_1^{rect}, (I_{c_1, max}, I_{c_2, max}, I_{c_3, max}) \equiv p_2^{rect}$ : the minimum and maximum intensity values, respectively, of color channels  $c_1, c_2$  and  $c_3$  (in %)
- $T \equiv p_3^{rect}$ : the period (in absolute time unit)
- $D_{rect} \equiv p_4^{rect}$ : the fraction of the period in which the signal is maximal

A rectangle waveform  $rect(t)$  on a color channel  $c_k$  is defined by:

<sup>6</sup> [https://github.com/adafruit/Adafruit\\_NeoPixel](https://github.com/adafruit/Adafruit_NeoPixel).

$$\text{rect}(t) = \begin{cases} I_{c_k, \max} & 0 \leq t < D_{\text{rect}} T \\ I_{c_k, \min} & D_{\text{rect}} T \leq t < T \end{cases} \quad (3)$$

$$\text{rect}(t) = \text{rect}(t + zT) \quad z \in \mathbb{Z}$$

#### 4.2.2 Triangle Waveform

The parameter vector  $\mathbf{p}_{\text{tri}}$  for this periodic waveform is composed of the following components:

- $(I_{c_1, \min}, I_{c_2, \min}, I_{c_3, \min}) \equiv p_1^{\text{tri}}, (I_{c_1, \max}, I_{c_2, \max}, I_{c_3, \max}) \equiv p_2^{\text{tri}}$ : the minimum and maximum intensity values for each color channel (in %)
- $T \equiv p_3^{\text{tri}}$ : the period (in absolute time unit)
- $D_{\text{tri}} \equiv p_4^{\text{tri}}$ : the ratio of the rise time to the period

A triangle waveform  $\text{tri}(t)$  on a color channel  $c_k$  is defined by:

$$\text{tri}(t) = \begin{cases} \frac{I_{c_k, \max} - I_{c_k, \min}}{D_{\text{tri}} T} t + I_{c_k, \min} & 0 \leq t < D_{\text{tri}} T \\ \frac{I_{c_k, \max} - I_{c_k, \min}}{1 - D_{\text{tri}}} \left(-\frac{t}{T} + 1\right) + I_{c_k, \min} & D_{\text{tri}} T \leq t < T \end{cases} \quad (4)$$

$$\text{tri}(t) = \text{tri}(t + zT) \quad z \in \mathbb{Z}$$

#### 4.2.3 Sinusoidal Waveform

The parameter vector  $\mathbf{p}_{\text{sinl}}$  for this periodic waveform is composed of the following components:

- $(I_{c_1, \min}, I_{c_2, \min}, I_{c_3, \min}) \equiv p_1^{\text{sinl}}, (I_{c_1, \max}, I_{c_2, \max}, I_{c_3, \max}) \equiv p_2^{\text{sinl}}$ : the minimum and maximum intensity values for each color channel (in %)
- $T \equiv p_3^{\text{sinl}}$ : the period (in absolute time unit)

A sinusoidal waveform  $\text{sinl}(t)$  on a color channel  $c_k$  is defined by:

$$\text{sinl}(t) = \sin\left(\frac{2\pi}{T}t - \frac{\pi}{2}\right) \frac{I_{c_k, \max} - I_{c_k, \min}}{2} + I_{c_k, \min} \quad (5)$$

#### 4.2.4 Modulated Waveform

This is a special kind of waveform combining several cycles from the three previous periodic signal shapes to create a “supercycle”. This “supercycle” is then periodically repeated. The parameter vector  $\mathbf{p}_{\text{modw}}$  for this periodic waveform is composed of the following components:

- $n_{\text{sub}} \equiv p_1^{\text{modw}}$ : the number of subcycles in one supercycle.
- $v_{\text{sup}} = (v_{\text{sub},1}, \dots, v_{\text{sub},n_{\text{sub}}}) \equiv p_2^{\text{modw}}$ : a vector of  $n_{\text{sub}}$  shape identifier - shape parameters pairs  $v_{\text{sub},i} = (\text{sh}_{\text{sub},i}, \mathbf{p}_{\text{sub},i})$ , describing the light behavior in each subcycle.

A modulated waveform  $\text{modw}(t)$  on a color channel  $c_k$  is defined by:

$$\text{modw}(t) = \begin{cases} \text{sh}_{\text{sub},1}(t) & 0 \leq t < T_1 \\ \vdots & \vdots \\ \text{sh}_{\text{sub},i}(t) & T_{i-1} \leq t < T_i \\ \text{sh}_{\text{sub},n_{\text{sub}}}(t) & T_{n_{\text{sub}}-1} \leq t < T_{n_{\text{sub}}} \end{cases} \quad (6)$$

$$\text{modw}(t) = \text{modw}\left(t + z \sum_{i=1}^{n_{\text{sub}}} T_i\right) \quad z \in \mathbb{Z}$$

where  $T_i$  represents the period of the  $i$ th subcycle.

#### 4.2.5 Step Function

The parameter vector  $\mathbf{p}_{\text{step}}$  for this periodic waveform is only composed of the following component:

- $(I_{c_1, f}, I_{c_2, f}, I_{c_3, f}) \equiv p_1^{\text{ramp}}$ : the final intensity value for each color channel (in %). The initial intensity value  $(I_{c_1, i}, I_{c_2, i}, I_{c_3, i})$  (previous state of the strip) is not relevant for animating the strip from  $t = 0$  onward, so it is not included in the parameter vector.

A step function  $\text{step}(t)$  on a color channel  $c_k$  is defined by:

$$\text{step}(t) = \begin{cases} I_{c_k, i} & t < 0 \\ I_{c_k, f} & t \geq 0 \end{cases} \quad (7)$$

#### 4.2.6 Clipped Ramp Function

The parameter vector  $\mathbf{p}_{\text{ramp}}$  for this periodic waveform is composed of the following components:

- $(I_{c_1, i}, I_{c_2, i}, I_{c_3, i}) \equiv p_1^{\text{ramp}}, (I_{1, f}, I_{2, f}, I_{3, f}) \equiv p_2^{\text{ramp}}$ : the initial and final intensity values for each color channel (in %)
- $t_{\text{rise}} \equiv p_3^{\text{ramp}}$ : the rise time (in absolute time unit)

A clipped ramp function  $\text{ramp}(t)$  on a color channel  $c_k$  is defined by:

$$\text{ramp}(t) = \begin{cases} I_i & t < 0 \\ \frac{I_f - I_i}{t_{\text{rise}}} t + I_i & 0 \leq t < t_{\text{rise}} \\ I_f & t \geq t_{\text{rise}} \end{cases} \quad (8)$$

For all of these animations, note that, in a Red–Green–Blue (RGB) color space ( $c_1 = R$ ;  $c_2 = G$ ;  $c_3 = B$ ), if the *color ratios*  $r_1$  and  $r_2$  listed below are constant as a function of time, then the animation is of a single color. If the ratio is not respected however, we would observe color changes

throughout the animation. Color ratios  $r_1$  and  $r_2$  are defined as:

- $r_1 \equiv I_{R,\max} : I_{G,\max} : I_{B,\max}$  and  $r_2 \equiv I_{R,\min} : I_{G,\min} : I_{B,\min}$  for the periodic signal shapes, and
- $r_1 \equiv I_{R,f} : I_{G,f} : I_{B,f}$  and  $r_2 \equiv I_{R,i} : I_{G,i} : I_{B,i}$  for the non-periodic signal shapes

The case where  $r_2 = 0 : 0 : 0$  results in a single color animation for any  $r_1$  values.

### 4.3 Animating a Digital RGB LED Strip

The animation model described in the previous sections is useful for design and visualization purposes. However, this model presents us with some limitations in practice when working with digital addressable LED strips. For the rest of this section, we assume we are working in a RGB color space.

#### 4.3.1 Light Animation as a Sequence of Frames

Most light strips nowadays are digital LED strips and therefore have a finite refresh rate  $f_{\text{refresh}}$ . This means that the intensity functions  $i_{jc}(t)$  ( $c = R, G, B$ ) are actually synchronized discrete signals  $i_{jc}[l] = i_{jc}(l \cdot \Delta t)$  where  $\Delta t = \frac{1}{f_{\text{refresh}}}$ . Moreover, the color levels are quantized (usually into 256 levels) for each Red, Green or Blue color channel. The combined discrete  $i_{jc}[l]$  values form the discrete-valued matrix  $\hat{A}[l]$ , which we call an *animation frame*.

**Definition 8** An *animation frame*  $\hat{A}[l]$ , at time step  $l$ , for a 256-level RGB digital LED strip containing  $n$  pixels is defined as:

$$\hat{A}[l] = \begin{pmatrix} i_{1R}[l] & i_{1G}[l] & i_{1B}[l] \\ \vdots & \vdots & \vdots \\ i_{nR}[l] & i_{nG}[l] & i_{nB}[l] \end{pmatrix} \tag{9}$$

where the rows represent the pixel indices and the columns represent the R, G and B color channels. The intensity values are discretized and quantized such that:

$$\forall l \in \mathbb{N} : i_{j,c}[l] \in \mathbb{N}, \quad 0 \leq i_{j,c}[l] \leq 255 \quad j = 1, \dots, n; \\ c = R, G, B$$

#### 4.3.2 Episodic Animation Control

The animations as described in the previous paragraph start at time step  $l = 0$  and extend arbitrarily in time. However, if these lights are going to be used to express a dynamic process, such as the varying state of a robot, then frequent switches from one animation to another will be needed, hence the notion of episodic animation introduced next.

**Definition 9** An *animation episode* is defined as a fixed portion of a particular animation. Animation episodes can be arbitrarily defined; for the signal shapes we considered, we define an episode to be:

- A single signal cycle, for periodic signal shapes, and
- The portion of the signal needed to transition from the initial to the final intensity value, for non-periodic signal shapes.

For a complete animation tuple set containing different animations for different pixels, each with its own episode, we define the complete animation episode to be lasting as long as the shortest of the individual episodes.

Based on the above definition, we propose an *episodic animation control algorithm*, which given a dynamic process DP of time-varying complete animation tuple sets, generates a sequence of animation episodes, as shown in Fig. 7. For illustration purposes, we assume that pixels 1 through  $n$  have the same animation, so we only show the corresponding animation tuple (the complete animation tuple set in this case is a singleton containing that tuple). The algorithm used to achieve this behavior is summarized in Algorithm 1. At the beginning of each episode, the algorithm gets the current animation tuple set value from DP and animates the strip with one episode of the animation corresponding to that tuple set. We assume that the choice of episode length is small compared to the dynamics of DP, in other words the minimum switching time for the animation tuple set values is greater than any single animation episode.

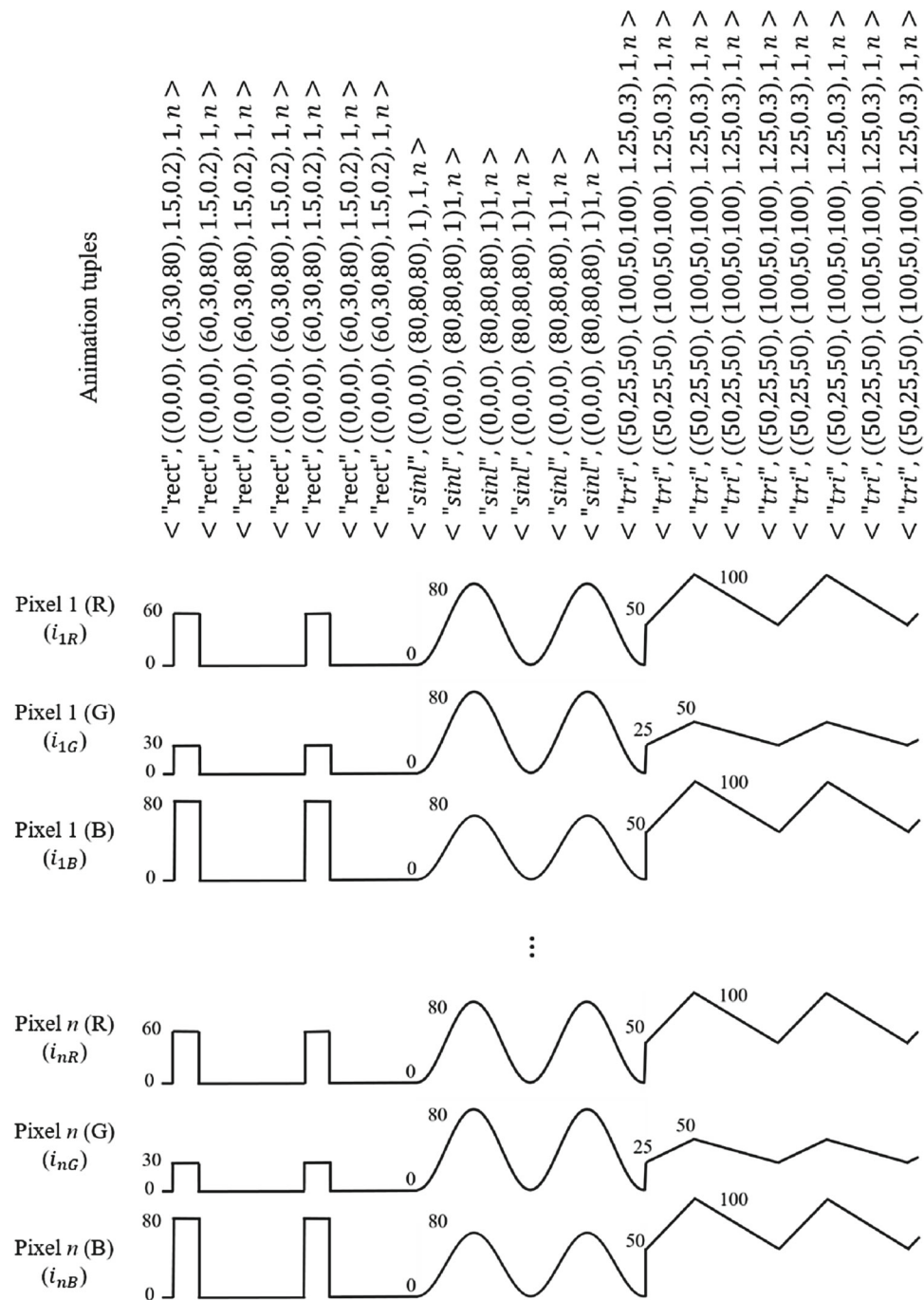
Let  $d_{\text{frame}}$  be the total delay between two animation frame updates.  $d_{\text{frame}}$  comprises several sources of delays including the time to compute the frame and the time to refresh the strip. Note that for a specific animation,  $d_{\text{frame}} = \Delta t = \frac{1}{\text{Refresh Rate}}$ . Since these delays can differ from frame to frame, it is best to keep track of time using a timer and sample the original  $A(t)$  curves at the corresponding time values, as shown in the “Episode” procedure of Algorithm 1, to get the corresponding discrete and quantized frames  $\hat{A}[l]$ , computed by the “GetFrame” procedure.

As a final note, we can easily extend our framework to more than one light strip to be independently animated. For  $N$  light strips, each will have its own dynamic process  $DP_i, i = 1, \dots, N$  to be animated. Let  $P_1, \dots, P_N$  be independent control process, one for each strip. Each  $P_i$  will be independently running Algorithm 1 on  $DP_i$  to episodically control light strip  $i$ .

#### 4.3.3 Section Summary

In this section, we presented a formal framework for representing and controlling light animations, summarized in

**Fig. 7** Episodic animation example: animation tuple as a function of time and corresponding intensity functions for each pixel of the strip



**Fig. 8.** We first introduced a representation of light animations as complete sets of animation tuples. We then focused on the control of RGB LED strips and present an episodic animation control algorithm to enable the translation of a dynamic process of complete animation tuple sets into a sequence of frames on a light strip. Our framework (Arduino code available online<sup>7</sup>) is not platform-specific and can be used by any device using our communication protocol.

<sup>7</sup> <https://github.com/kobotics/LED-animation>.

## 5 Robot State/Animation Mapping: Formalism

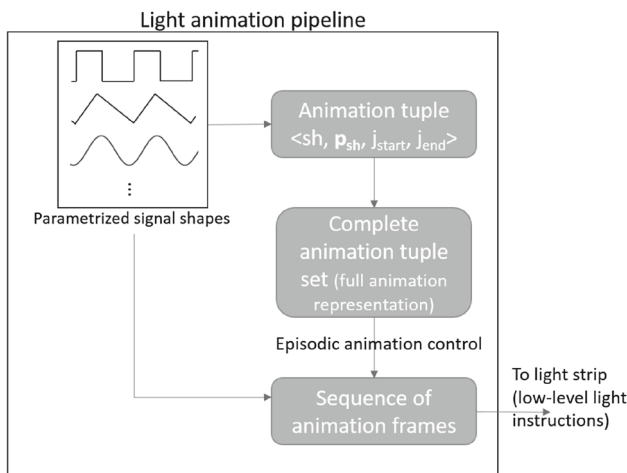
Now that we have discussed our representation of the robot state both on its own and in relation to an expression channel, we look at the problem of mapping expressible classes to specifically light animations, whose formalism was presented in Sect. 3.

**Algorithm 1** Episodic animation control algorithm  
RunAnim

```

1: procedure RUNANIM(DP)
2:   tuplenow ← NULL
3:   StartTimer(t) ▷ t globally stores the time elapsed in seconds
4:   while true do
5:     tupsetprev ← tupsetnow
6:     tupsetnow ← FetchAnimTuple(DP)
7:     if tupsetprev ≠ tupsetnow then ResetTimer(t)
8:     Episode(tupsetnow)
9:   procedure EPISODE({{sh1, psh1, jstart1, jend1}, ...,
    {shm, pshm, jstartm, jendm}})
10:    {e1, ..., em} ← GetEpisodeLengths(psh1, ..., pshm)
11:    emin ← mini ei
12:    while t < emin do
13:      Â ← GetFrame({{sh1, psh1, jstart1, jend1}, ...,
        {shm, pshm, jstartm, jendm}}), t)
14:      UpdateStrip(Â) ▷ Updates the strip with the current frame
15:    procedure GETFRAME({{sh1, psh1, jstart1, jend1}, ...,
      {shm, pshm, jstartm, jendm}}), t)
16:      for i ← 1, ..., m do
17:        for j ← starti, ..., endi do
18:          Âj,1:3 ← (round( $\frac{255}{100} \cdot i_{j,R}(t)$ ), round( $\frac{255}{100} \cdot i_{j,G}(t)$ ),
            round( $\frac{255}{100} \cdot i_{j,B}(t)$ ))

```



**Fig. 8** Summary of the animation framework described in this section

**5.1 Mapping Architecture**

The mapping we define between expressible classes (state information) and animation tuples (light animation information) is divided into two parts:

- Animation determination: The signal shape *sh* as well as the default parameters **psh** and *j<sub>i</sub>* (*i* = start, end) are determined by the ‘feature’ part of expressible state tuple, which is by definition discrete and communicates well the presence of distinct robot states.
- Animation modulation: We allow modulation in the animation by modifying the value of the parameters (for a

fixed signal shape) based on the value(s) of the ‘variable’ part of the expressible state tuple.

Figure 9 summarizes the mapping architecture, and Fig. 10 shows the flow of the mapping process for robot state all the way to the light animation module.

**5.2 Expression of Non-exclusive State Features**

As can be observed in the way state features are defined, there is no constraint on their mutual exclusivity, which means that there could be two or more features which are true at the same time, and hence more than one expressible tuple competing to be expressed on the same channel. If we assume that only a single expressible state tuple can be expressed on a single channel then we need a way of selecting one out of the set of expressible state tuples, which we call state preference function.

**Definition 10** A state preference function for a given expression channel is a function  $\phi : P(F) \rightarrow F$  where *F* is the set of state features and *P(F)* is the power set of *F*.  $\phi$  selects one preferred state feature given a set of true state features.

In practice, there might be sets of mutually exclusive features, which can reduce the domain of  $\phi$ . Also, it might be possible to create a strict total order on the state features, which greatly simplifies the representation of  $\phi$ , as is shown in the example below, but it might generally not be the case.

**Example of preference ordering on sample features:** Consider the three features ‘Escorting’ = *f<sub>1</sub>*, ‘Blocked by an obstacle’ = *f<sub>2</sub>*, and ‘Asking for help at an elevator’ = *f<sub>3</sub>*.

If the robot is blocked by an obstacle (*f<sub>2</sub>* is true), then it should express it even if it currently escorting a person (*f<sub>1</sub>* is true). (*f<sub>2</sub>* and *f<sub>1</sub>* are mutually exclusive since the robot cannot be blocked by an obstacle while it is statically waiting at an elevator).

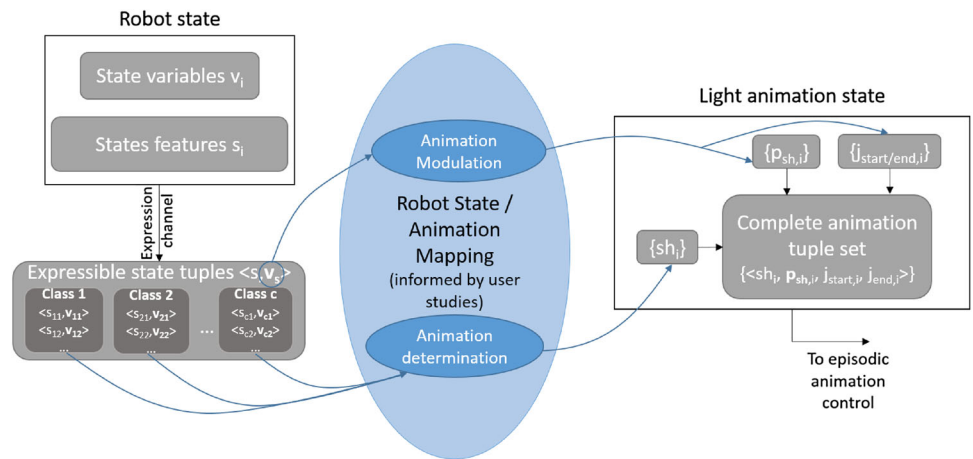
Similarly, if the robot is escorting a person (*f<sub>1</sub>* is true) but across multiple floors, it will encounter situations where it is asking for help at an elevator (*f<sub>3</sub>* is true) while performing the escort task. In such situation, we prefer *f<sub>3</sub>* over *f<sub>1</sub>* since the service cannot continue if the ‘ask’ task is not completed, which is hence more important.

Therefore we have the following total order: *f<sub>2</sub>* > *f<sub>3</sub>* > *f<sub>1</sub>*, where *x* > *y* indicates that *x* is preferred over *y*.

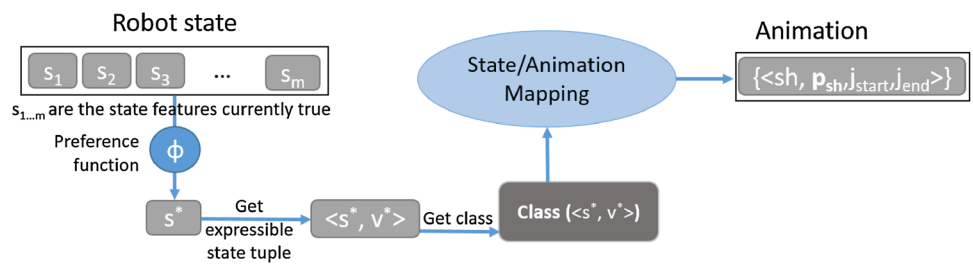
**5.3 Extension to Multiple Light Strips/Expression Channels**

The mapping method introduced above for a single channel can easily be extended to handle multiple expression channels. Even though our analysis focuses on two channels of the same modality (expressive lights), the architecture presented

**Fig. 9** Architecture of the robot state/animation mapping



**Fig. 10** Flow diagram of the robot state/animation mapping



is general enough to be applied to other modalities such as speech, expressive motion, etc.

5.3.1 Mapping Architecture

Figure 11 shows our mapping architecture when more than one light strip, or more generally more that expression channel, is present. Each channel has their relevant state features (possibly repeated), preference function, expressible state tuples and expressible classes.

5.3.2 Example Using Two Light Strips for Multi-level Expression

To demonstrate the extensibility of our concept to more than one expression channel, we consider one light strip for expression of higher-level state information (related to tasks and services), and another light strip for expression of lower-level state information (related to navigation parameters). Implementation details on hardware mounting and animation control can be found in the next section. The two strips can be seen in Fig. 12. The high level strip animations will be informed by user studies in the next section. For the low-level strip, we considered a turn indicator animation, which lights either the left or the right part of the strip depending on the turn the robot is about to take. The rest of the strip changes color as a function of speed (red for fast, orange for medium speed and green for low speed). The corresponding expressible tuple

for this strip’s expressive behavior is:  $\langle \text{‘Turning’}, (\text{next-turn}(\text{loc}, \text{path-plan}), \text{speed}) \rangle$ .

5.4 Implementation of the Mapping on a Real Robot

The framework discussed above for mapping robot state to light animations has been implemented on one of our CoBots and has been robustly running for more than a year whenever the robot is deployed. In this section we provide some details about the hardware and software components used to robustly integrate the light expressive behavior on CoBot.

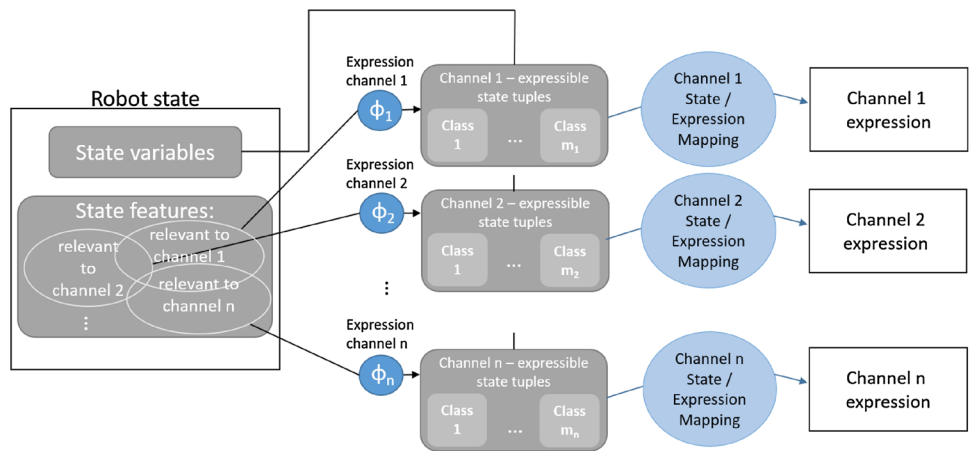
5.4.1 Hardware Components

For our light sources, we used two programmable, fully addressable NeoPixel LED strips<sup>8</sup> with 91 and 144 pixels respectively mounted on the robot’s body and around its base respectively, as can be seen in Fig. 13. Acrylic diffusers were added around the body LED strip to achieve omnidirectional visibility. Compared to other options like luminous fabrics or LED panels, linear strips are both simple in structure and flexible to adopt different mounting alternatives on CoBot. The NeoPixels strip moreover provides high light intensity thanks to its density of 144 LEDs/m (35 Watts/m max) which makes it suited for good visibility in luminous areas such as indoor bridges or other areas with glass windows.

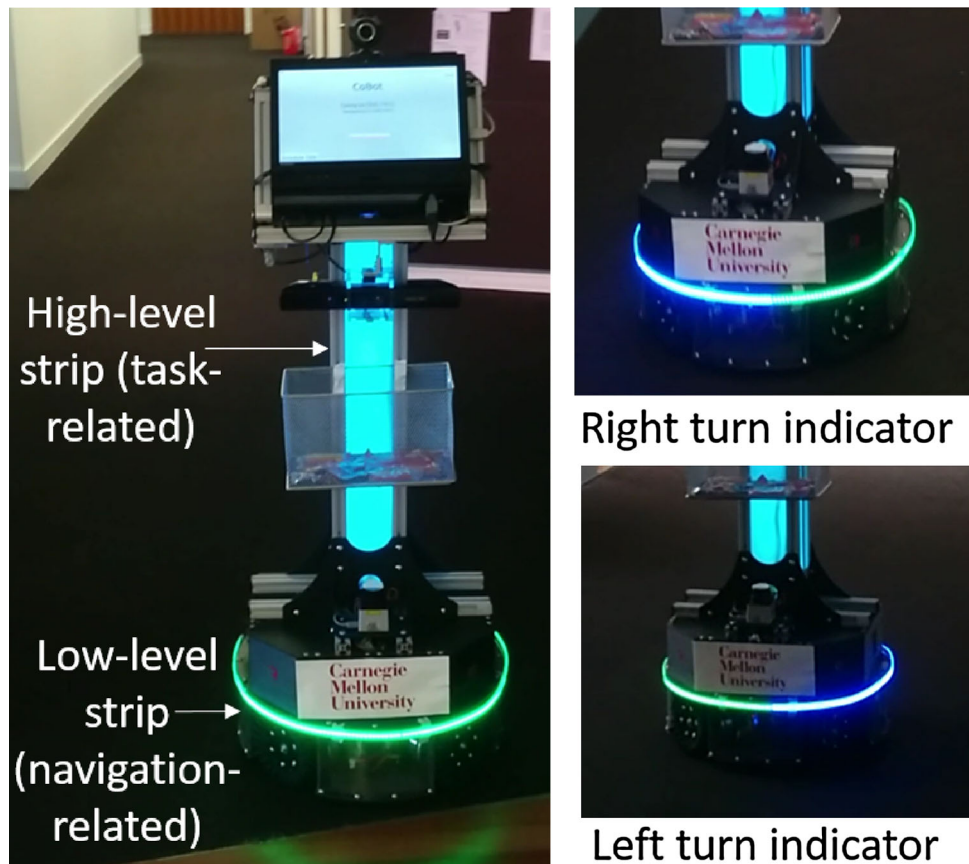
<sup>8</sup> <https://www.adafruit.com/products/1507>.



**Fig. 11** Mapping architecture for multiple expression channels



**Fig. 12** Two expressive light channels for multi-level expression on CoBot (left) and turn indicator snapshots for the low-level strip

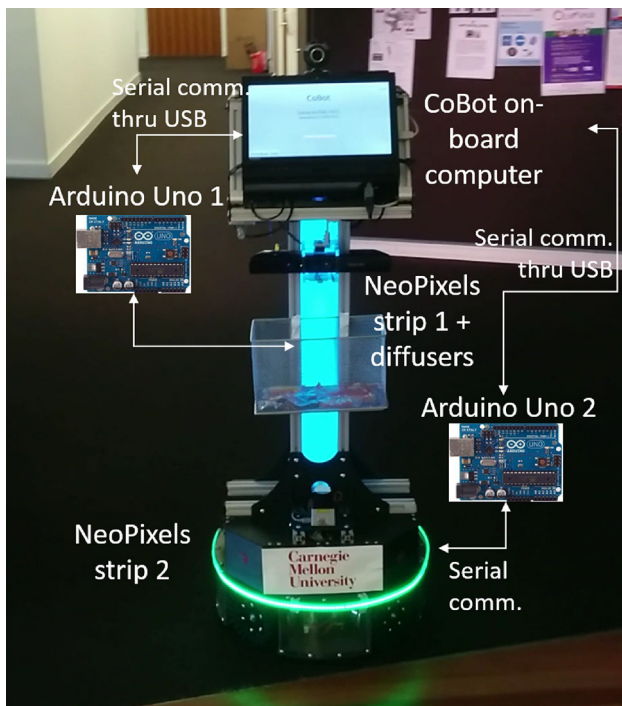


The light strips are controlled by Arduino Uno micro-controllers<sup>9</sup> (one per strip). The data pin of the strips are connected to a digital output pin of the corresponding Arduino. The Arduino in turn is connected to the CoBot on-board computer through a USB cable used for serial communication.

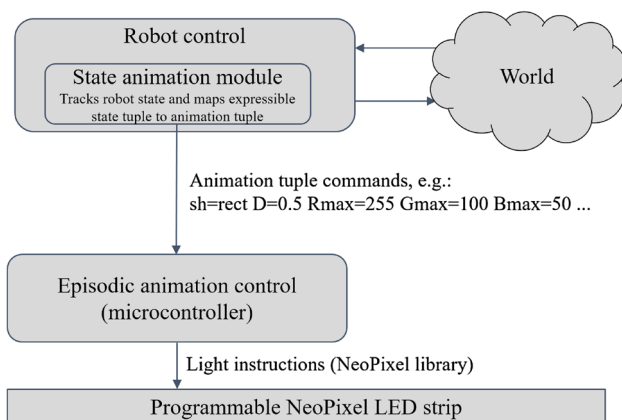
5.4.2 Control Architecture

The light interface control architecture is summarized in Fig. 14. A Robot Operating System (ROS) node running on the robot itself keeps track of the robot variables (by subscribing to the corresponding ROS topics or calling the corresponding ROS services) and computes the corresponding expressible state tuples. It then maps these to animation tuples for each strip, which are sent to the corresponding microcontroller using serial communication. The protocol used for communication between the ROS node and the

<sup>9</sup> <http://store-usa.arduino.cc/products/a000066>.



**Fig. 13** Hardware interface design



**Fig. 14** Control diagram of the state animation interface

microcontroller is available online.<sup>10</sup> Based on the animation tuples received, the microcontroller controls the light strip by running the episodic animation algorithm described in Sect. 4. The flow of data from the ROS node to the microcontroller is synchronized with the animation episodes because serial communication is only reliable when the microcontroller is not sending any data out to the strip. We ensure such synchronization by a simple procedure similar to a handshake at the end of each animation episode. The actual data sent out to the LED strip on the digital pin of the Arduino

is determined by the Adafruit NeoPixel library<sup>11</sup> and uses serial packets that the WS2811-based pixels understand.

## 5.5 Section Summary

In this section, we showed how to map the extracted elements of the robot's state to an expressive behavior, in particular a light animation as defined in the previous section. We extended our method to handle more than one expression channel, which is illustrated using two light strips on CoBot expressing different types of state information (high-level task-related versus low-level navigation-related information). Our process, although it has focused on CoBot, is general enough to apply to other platforms, and other expression channels than lights.

## 6 Design and Evaluation of the State/Animation Mapping

The previous section was concerned with question of *what* to express, i.e., selecting appropriate state information that is useful and relevant for a given situation. In this section, we focus on the question of *how* to express the selected state information. In order to answer this question, we present three studies. The first study [2] is concerned with the design of appropriate animations for some of the expressible state tuples discussed in Sect. 3, i.e. selecting the appropriate animation parameters discussed in Sect. 4 according to the different scenarios considered. The second study [3] evaluates the *legibility* of the animations resulting from our design study, as well as their generalizability to expression of state tuples in the same class (refer to Sect. 3). While the first study uses feedback from fully informed experts, the participants of the second study were non-experts who were only given minimal information about the testing scenarios. The third study is a small experiment which proves that the presence of these animated lights on the robot can actually influence people's behavior to help the robot perform better at its tasks.


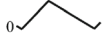
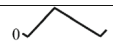
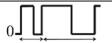
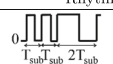

### 6.1 User Study 1: Designing Appropriate Animations

In order to select suitable parameters for the animations presented above, we conducted a study with a video-based survey. Participants were given a detailed description about three different scenarios involving CoBot and possibly a human. There were then instructed (through email) to watch videos of the robot in each of the scenarios, on which they had to report answers through the form of a provided spreadsheet.

<sup>10</sup> <https://github.com/kobotics/LED-animation>.

<sup>11</sup> [https://github.com/adafruit/Adafruit\\_NeoPixel](https://github.com/adafruit/Adafruit_NeoPixel).

**Table 2** Parameter values for the animation choices shown

Scenario "Waiting"			Scenario "Blocked"			Scenario "Progress"	
<i>wv</i>	<i>D</i>	<i>T</i> (s)	<i>wv</i>	<i>D</i>	<i>T</i> (s)	<i>disp</i>	<i>u<sub>disp</sub></i>
Blink			Push				
	0.5	2/1.6/0.6		0.25	1.5/1/0.5	prog_bar	bottom_up
Siren			Aggressive Blink				
	0.5	2/1.6/0.6		0.5	2/1.6/0.6	prog_bar	top_down
Rhythmic Blink			Color change				
	0.5	3/2.5/1.5		1	-	color_change	-

### 6.1.1 Preliminary Study

A preliminary study was conducted with the people who have the most expertise for our purposes, namely the CoBot developers. Eight developers participated in the survey, and submitted their choices. To validate our design choices, we recruited 30 more people to include in the study. The results across both studies were consistent. The extended study is described next.

### 6.1.2 Participants

A total of 38 participants took part in this study. Our recruitment filter only included participants who have expertise in one of the following fields: robotics (61% of the participants), design (18% of the participants), and engineering (21% of the participants). The participants ages ranged from 19 to 50, with an average of around 25 years old. Because cultural differences may have an impact on our results, we tried to have diversity in our sample: 18% of the participants are from North America, 32% from Europe, 29% from the Middle East and 21% from Asia. Additionally, 68% were male and 32% female.

### 6.1.3 Survey Design

Participants were asked to give their input on three aspects of the animation: animation pattern, speed and color. For each scenario, a single video sequentially showed 3 different animation patterns (corresponding to signal shapes + dynamics parameters) with the same neutral color (soft blue). Nuances of 3 different speeds were also shown within each pattern. The participants were prompted to select the (animation pattern, speed) pair that they thought would fit best the robot's expression purposes in the given scenario, of which they were fully informed. In the last section of the video, we showed 6 possible light colors (in the form of a static image of the robot). The participants were also asked to report on the color they thought was most appropriate for their selected (animation pattern, speed) choice, for each scenario. We made the reasonable assumption that the choice of color for the animation is independent of the actual animation selected, which

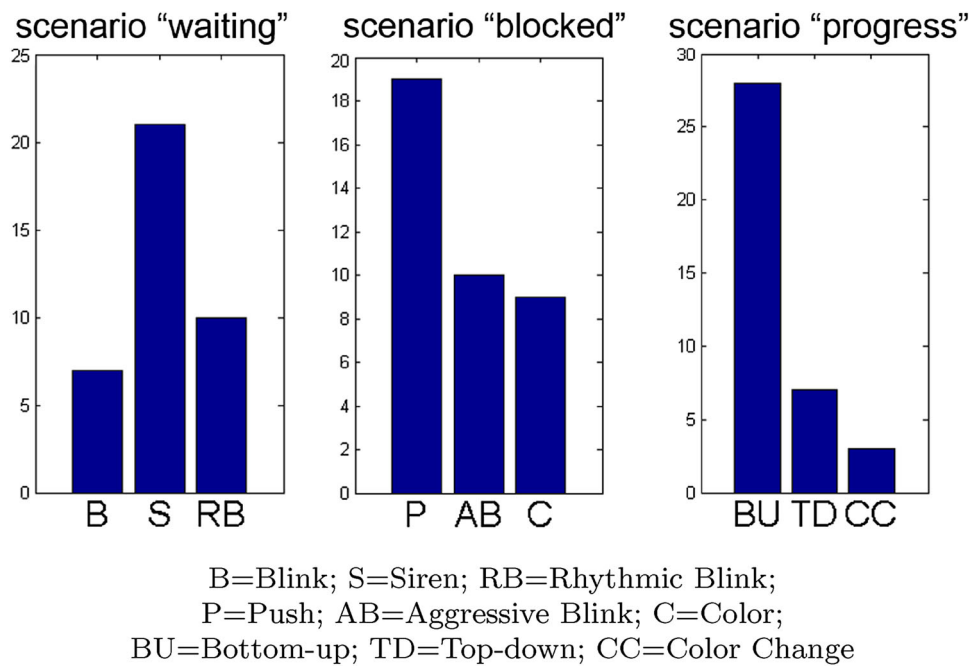
helps reduce the amount of choices to be shown. Indeed, while animation pattern and speed both relate to modulations in time and intensity, color seems to be much less intertwined to the other two. Furthermore, according to color theory [35], color on its own plays a strong role in expression. Next, we list and justify the choices of animation patterns shown to the participants.



- Scenario "waiting": A regular blinking animation (Blink); a siren-like pattern; a rhythmic (non-regular) blinking animation. We believe these to be good candidates for grabbing attention because of the dynamic aspect, the warning connotation and the non-regular pattern respectively.
- Scenario "blocked": A faded animation (that we call "Push") that turns on quickly and dies out slower (giving the impression of successively pushing against an obstacle); an "aggressive" blink (fast blink followed by slow blink); a simple color change at the time the robot gets blocked. We believe these to be good candidates for inciting the human to move away from the path.
- Scenario "progress": A bottom-up progress bar where lights gradually fill from top to bottom proportionally to the distance from the goal; a top-down progress bar where lights fill from the top towards the bottom; a gradual change from an initial color to a final color, again proportionally to the distance from goal.

The parameter values associated with these animations are summarized in Table 2. In addition to the animation summarized in the table, the following colors were shown for each scenario as static images of the lighted robot: Red (R), Orange (O), Green (G), Soft Blue (B), Dark Blue (B') and Purple (P).

### 6.1.4 Results

Figure 15 and Table 3 show the distribution of the results in the extended study. In the following discussion, *p*-values are obtained from a  $\chi^2$  goodness-of-fit test against a uniform distribution. Table 3 shows the selected best choices, which

**Fig. 15** Animation pattern results**Table 3** Selected best animations for each scenario of user study 1

Scenario	Animation and parameters			
“Waiting”	Soft blue “Siren” with period 2 s			
	$wv$	$D$	$T$ (s)	Color
		0.5	2	Soft Blue
“Blocked”	Red “Push” with period 1.5 s			
	$wv$	$D$	$T$ (s)	Color
		0.25	1.5	Red
“Progress”	Bottom-up progress bar			
	$disp$	$u_{disp}$	In. color	Fin. color
	prog_bar	bottom_up	Red	Green

were consistent between the preliminary and the extended study.

In Fig. 15, we show the results for the animation pattern. For the scenario “waiting” ( $p = 0.0137$ ), among the participants who chose the winning animation “Siren”, 64% chose the slower speed, 29% the medium speed and 7% the faster speed. For the scenario “blocked” ( $p = 0.0916$ ), among the participants who chose the winning animation “Push”, 27% chose the slower speed, 40% the medium speed and 33% the faster speed. Note that the static color change was the least preferred animation pattern for this scenario, which aligns with Bertin’s result stating that motion (in our case a non-static animation) being one of the most effective visual features for attention grabbing [4] (for the “waiting” scenario which relies even more on attention grabbing, all three of our designed patterns were designed to be in motion). For

**Table 4** Color results (color codes correspond to those used in the text)

R (%)	O (%)	G (%)	B (%)	B' (%)	P (%)
Scenario “waiting”					
13	13	13	39	16	6
Scenario “blocked”					
53	29	5	0	10	3
R/G (%)	B/P (%)	B'/G (%)	O/G (%)	O/B (%)	P/B (%)
Scenario “progress” (top 6)					
27	12	12	8	8	8

the scenario “progress” ( $p = 1.10^{-6}$ ), the participants chose the bottom-up progress bar animation. All  $p$ -values obtained are below 0.10, which indicates a strongly non-uniform distribution of preferences for each scenario, and this can clearly be seen in Fig. 15.

The results for colors, summarized in Table 4 similarly show a clear preference for one option in each case. For instance, soft blue was selected for the “waiting” scenario. This result supports the statement in [7] that cold colors are better than warm colors at grabbing attention. Also, red was selected as the best color for the “blocked” scenario. This is consistent with the fact that red is often perceived as demanding [35] or stimulating [7], which are both desirable in this scenario. Even though the Red/Green color combination was the most voted for in the “waiting” scenario, this study did not account for interference between animations for different scenario. As it turns out from real-world deployments, since

the color red was associated with the “blocked” scenario, it confused people to also see it in the “progress” scenario. Also, since Red–Green color blindness is the most widespread type of colorblindness, it is a good design principle to avoid this combination of colors in a single animation. For these reasons, we opted to replace the background color with a faint dark blue, while preserving the bright green progress bar.

### 6.1.5 Discussion

The results of the study demonstrate strongly non-uniform selected choices on our (non-randomly) designed light animations, which suggests that there are some semantics associated with simple animations of a colored light strip, in relation to a concrete robot state. From a design choice point of view, the results of this study suggest that we can safely eliminate choices which received a poor rating, resulting in a small set which can be considered valid. Although, for each scenario, there is generally a clear preference for one of the choices, visualized in Fig. 16, the study was informative of the distribution of preferences, which can even enable the possibility of generating animations in a probabilistic way, especially if more information about the human interacting with the robot is available (e.g., cultural background).

Moreover, the scenarios we looked at are quite generic, and are commonly encountered in interactions involving a social robot and a human. However, before extrapolating our results to other platforms, we need to ensure that other factors (e.g. strip size or placement, light diffusion mechanism...) do not influence the perception of the expressive behavior. The distinguishing feature of our robotic platform is motion, which prompts us to think whether other mobile robotic systems such as autonomous cars or drones could utilize similar animations if they encounter similar states. The main research questions to investigate to study generalizability at this scale (beyond the physical factors mentioned above) would concern the effect of the context on the perception of the task and the signals associated with it. For instance, would red and green colors mean different things in an indoor environment versus in a traffic scenario, or an aerial scenario, where existing visual codes might influence the perception? These results can however still serve as a starting point for the design of future social robotic systems which use lights as a means of communication.

## 6.2 User Study 2: Evaluating and Generalizing the Designed Animations

In order to evaluate the effectiveness of the chosen expressive light animations, we conducted an online survey in which participants watched videos of a robot performing tasks from afar. At the end of each video, participants were

asked to hypothesize about the robot’s current *state*, but also about its *actions* (specifically reasons for performing a specific action such as stopping or being unresponsive). Questions were in a multiple choice format, with four possible answers. Half of the participants saw the robot performing tasks with its expressive lights on (referred to as the “Lights on” condition), and half saw the robot with the lights off (referred to as the “Lights off” condition). The animations for the “Lights on” condition were informed by the first user study described in Sect. 6.1. Participants were randomly assigned to one of the two experimental conditions, resulting in a between-subject study. We analyzed participants’ hypothesis choices to demonstrate that those who saw the robot with the lights were more accurate, but also and gained a higher level of trust in robots, from watching the videos.

### 6.2.1 Participants

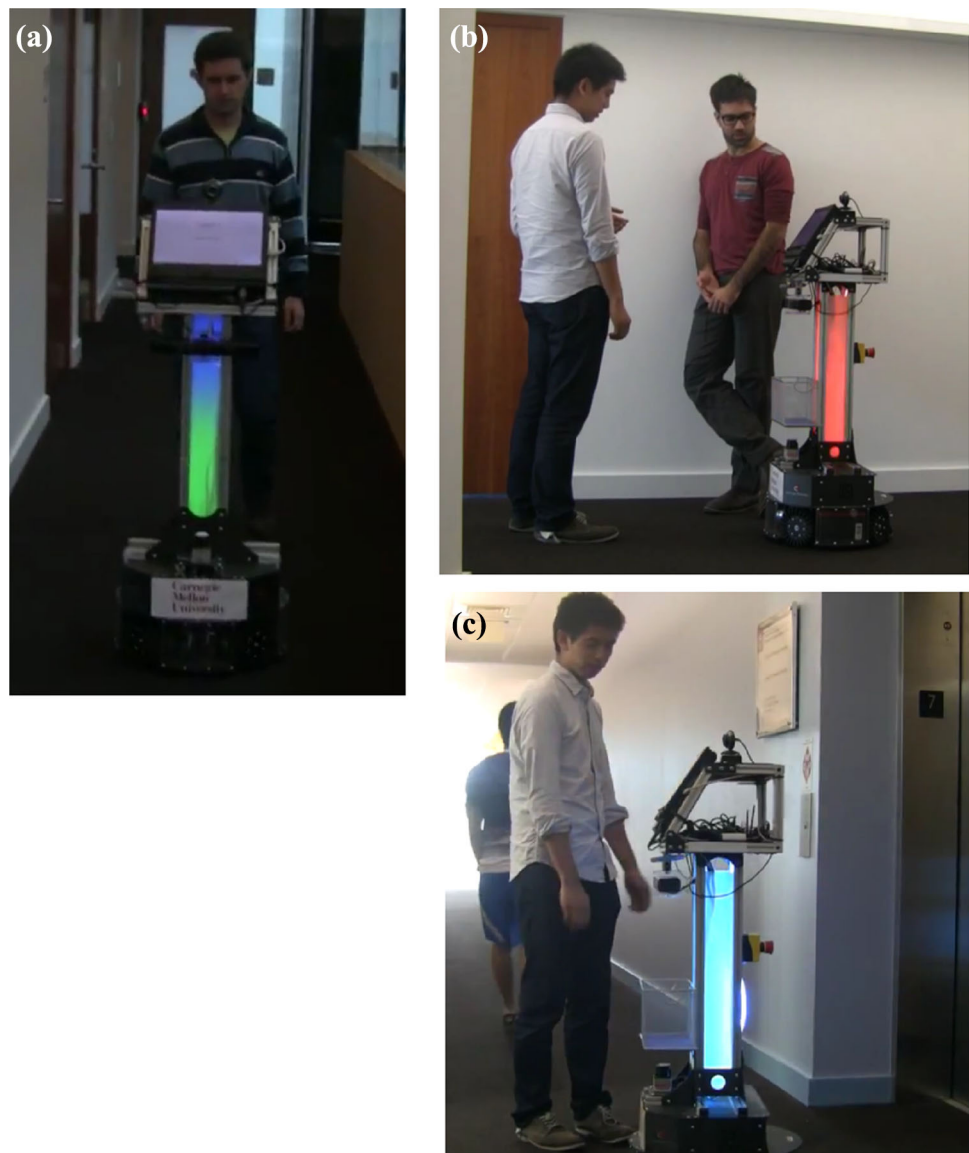
A total of 42 participants, of which 14 were male and 28 were female, took part in the study. Ages ranged from 20 to 67 ( $M = 32.4$ ,  $SD = 13.7$ ). Out of the 42 participants, 33 live in the United States; the rest live in different countries across Asia and Europe. Even though computer usage was relatively high amongst participants (31 out of 42 used computers 30+ hours per week), experience with robots was generally low. Only 5 out of the 42 participants reported having worked with robots before, and 20 reported that they have never seen a robot in person before (3 participants had seen our particular robot, CoBot, before taking the survey). Finally, we ensured that none of the participants were colorblind, since our light animations included color and it could have an effect on our results.

### 6.2.2 Survey Design

Our online video-based survey (running on the Limesurvey platform) comprised nine video scenarios of CoBot acting in our environment followed by a multiple choice question asking participants to choose a hypothesis of what the robot was doing. Four plausible hypotheses about the robot’s state/actions were presented as choices for each video, of which one had to be selected. Participants were also asked to rate their confidence in their answer on a 5-point Likert scale. The video order, as well as the choices for each answer, were randomized to avoid any order effects.

Each of the video scenarios was recorded using our autonomous robot with lights on and lights off. Some videos involved humans, which were actors behaving naturally and according to the specifications of the scenario. Although the robot was acting autonomously, the videos were replicated as close as possible for the two conditions. From the robot’s perspective, we can reasonably assume that the only notable

**Fig. 16** Snapshots of the winning animations for each scenario of user study 1, also used in the subsequent user studies. **a** Green ascending progress bar on an escort task, **b** flashing red “push” animation for path obstructions, **c** slow soft blue “siren” to call for human help. (Color figure online)



difference between the two videos for a given scenario is the presence or absence of lights on the robot. The videos did not include any robot speech or any visible information on the robot’s screen.

After viewing all nine videos, some relevant background and related information, including trust questions about this particular robot and robots in general, was also collected. Additionally, we recorded the time taken for completing the survey and made sure everyone responded within reasonable time limits (no disqualifications).

### 6.2.3 Scenario Descriptions

The nine scenarios shown in the videos were specifically chosen based on actual tasks that the robot performs while it is deployed in our buildings. We focused our scenarios

on the same three common *scenario classes* studied in our prior work – “progressing through a process”, “blocked”, and “waiting for human input”. For each scenario class, we produced three distinct scenarios in which the robot’s state or actions are ambiguous, which are summarized in Table 5 and described below.

The “**progressing through a process**” scenarios represent the robot taking actions for a long duration. For each of these scenarios, the progression was modeled as the light animation of a progress bar (see Subsection 1). The scenarios chosen to represent this class are:

- *Navigation task with human presence (P1)* A person participates in the Escort Task in which they are accompanied to their goal location.

- *Speech task (P2)* The person asks a question to the robot, which provides no immediate answer, as it is searching the web for the required information. The video ends before the robot responds. When present, the lights show the progress on the web query task.
- *Charging (P3)* The robot is charging inside the laboratory, with no clear view of the power plug. When present, the lights show the battery level increasing progressively (video sped up 10 times).

The “**blocked**” scenarios represent the robot being interrupted in its navigation by obstacles of different kinds. The important blockage is supported by the fast red flashing light (see Subsection 1). The scenarios chosen to represent this class are:

- *Human obstacle facing the robot (B1)* The robot is stopped in its navigation by a person standing in a narrow corridor, facing the robot.
- *Human obstacles looking away from the robot (B2)* The robot is stopped in its navigation by a person standing in a narrow corridor, facing away from the robot.
- *Non-human obstacle (B3)* The robot, navigating down a narrow corridor, detects a person walking towards it and changes its navigation path to avoid the person. As a result, it finds itself in front of a branch of plant, which it considers as an obstacle, causing it to stop.

The “**waiting for human input**” scenarios represent the robot stopped waiting for different types of actions to be taken by a human. For each of these scenarios, the robot is waiting patiently as represented by the slow flashing blue light (see Subsection 1). The scenarios chosen to represent this class are:

- *Waiting for help at an elevator (W1)* The robot is stopped in front of the elevator, waiting for someone to press the elevator button and let it in. People are passing by, ignoring the robot’s presence.
- *Object loading (W2)* The robot is stopped in the kitchen area, facing a counter on which we can see a cup of coffee. Next to the counter area, a person is washing the dishes, presumably unaware of the robot’s presence.
- *Confirming task completion (W3)* The robot is stopped in front of an office door, with coffee in its basket. A person shows up from inside the office and takes the coffee. The robot doesn’t react to the person’s action and remains still. The person looks at the robot with a confused look on their face.

For each scenario, when lights are present, the default animation on the robot (when no expression is desired) is a static soft blue color.

#### 6.2.4 Multiple Choice Questions

After viewing each video, the participants were given choices to explain the robot’s state or actions. As discussed earlier, each of the scenarios can be ambiguous to a person viewing CoBot from afar either because of lack of contextual information or because of mixed signals in the robot’s behavior. The corresponding answer choices for each video scenario were specifically chosen to reflect many of the possible hypotheses that could correspond to the robot’s behaviors. Given our prior work, we theorize that the light animations will reduce the uncertainty that people have in understanding robot’s behavior, leading to more accurate answers to our multiple choice questions.

##### Question examples

Some examples of questions and choices in the survey are:

*In the video above, why did the robot stop?* (a) The robot recognizes the person, who was expecting it, (b) The robot sees the person as an obstacle, (c) The robot needs help from the person, (d) The robot is inviting the person to use its services. (Scenario B1)

*In the video above, why is the robot not moving after the person has taken the coffee?* (a) It is waiting for the person to confirm the task is over, (b) It has nothing to do, (c) It is low on battery, (d) It is trying to get inside the room but the door is too narrow. (scenario W2)

#### 6.2.5 Results

Responses to the survey multiple choice questions in the nine scenarios were coded in a binary fashion—three answers were coded as wrong and one answer was coded as the correct answer. The resulting dependent variable *accuracy* was modeled as binary categorical. Additionally, we coded the responses to our questions about robot *trust* (5-point Likert scale). We analyzed the effects of our independent variables—experimental *condition* (binary categorical variable “Lights on” and “Lights off”) and *scenario* (nine categories)—on the dependent variables. While our scenarios had a range of difficulty resulting in a range of accuracies, our light animations have a statistically significant effect across all scenarios on participant’s accuracy. The participants who saw the robots with lights on also indicated an increase in their overall trust in robots more than those who saw the robot with lights off.

**Participant Accuracy** In order to analyze our categorical dependent variable *accuracy*, we used a McNemar’s  $\chi^2$  test in a combined between- and within-subject design. The “Lights on/off” *condition* is our between-subject variable. All nine video *scenarios* were shown to all participants and therefore is a within-subject variable. The *participant* is mod-

**Table 5** Scenarios used in user study 2

Scenario class	Progress to a goal (P)	Blocked (B)	Waiting for human input (W)
Scenario 1	Navigation task with human presence (P1)	Human obstacle facing the robot (B1)	Symbiotic autonomy (elevator button) (W1)
Scenario 2	Speech task (P2)	Human obstacles looking away from the robot (B2)	Object loading (W2)
Scenario 3	Battery charging (P3)	Non-human obstacle (B3)	Confirming task completion (W3)

eled as a random variable within the model as each person may be more or less accurate in general. The McNemar's  $\chi^2$  tested whether the participants' answers depend on the presence/absence of lights, video scenario, and/or the interaction effects of both the lights and video scenario together.

Our results indicate that there is a statistically significant difference in the accuracy based on the presence/absence of lights ("Lights on"  $M = 75.66\%$ ,  $SD = 18.20\%$ ; "Lights off"  $M = 56.08\%$ ,  $SD = 19.16\%$ ,  $p < 0.0007$ ). The accuracy was significantly higher for participants who saw the lights. Additionally, there is a statistically significant difference in participants' accuracy based on the video scenario (see Fig. 17 for means and standard deviations,  $p < 0.0001$ ) (i.e., some videos were harder to determine the robot's state/actions than others for each participant). However, there was no statistically significant effect by the interaction of the light condition and the video scenario ( $p = 0.7$ ), indicating that the increased effectiveness of the "Lights on" condition was the same across scenarios. Based on these results, we conclude that while the choice of a correct robot state/actions hypothesis does depend on the scenario in which humans see the robot, the tested light animations universally help increase their accuracy.

Figure 17 shows the average accuracy of the participants for each scenario and each light condition. The error bars represent a 95% confidence interval of the mean. We note that the "Lights on" condition accuracies (shown in blue) are universally higher than the "Lights off" accuracies (shown in red). Additionally, the graph clearly shows our result that the video scenarios have different average accuracy, but the accuracy change between conditions per video scenario is not reflective of the scenario.

**Participant Trust in Robots** On average, participants reported that their trust in robots had increased after watching the videos shown in the survey. (To the question: "Do you agree with the following statement? 'After watching these videos, I will not trust robots as much as I did before.'", participants in both conditions answered above 3 over 5 on average on a 5-point Likert scale, where 1 meant "Strongly Agree" and 5 meant "Strongly Disagree".) The reported increase in their trust in robots was significantly more pronounced for participants in the "Lights on" condition ( $M =$

4.29,  $SD = 0.90$ ) compared to those in the "Lights off" condition ( $M = 3.52$ ,  $SD = 0.87$ ) ( $t(40) = \pm 2.02$  two-tailed,  $p = 0.008$ ).

However, there was no statistically significant difference between the two conditions in the reported absolute level of trust in both CoBot and in robots in general ( $t(40) = \pm 2.02$  two-tailed,  $p > 0.05$ ), only in the change in trust discussed above did the results differ across conditions.

**Participant Confidence** We analyzed the average reported answer confidence over the 9 videos and found that it was slightly higher in the "Lights on" condition as compared to the "Lights off" condition (2.9 vs. 3.17 respectively on a 5-point Likert scale). However, this difference was not statistically significant ( $t(40) = \pm 2.02$  two-tailed,  $p > 0.05$ ). Similarly, no statistical significance was found when comparing reported confidence on a per video basis.

### 6.2.6 Discussion

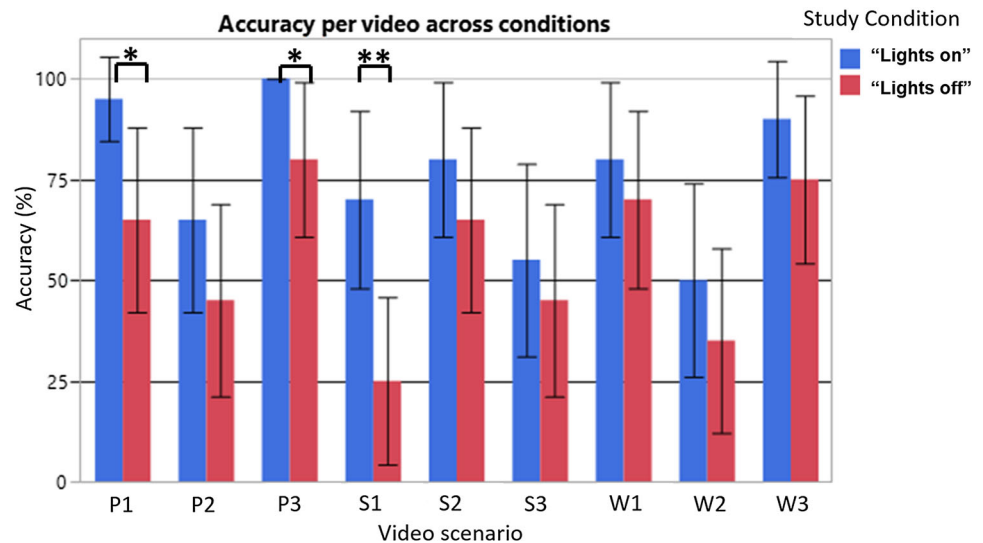
The significant effect of scenario on response accuracy shows that some questions were harder than others. This was not unexpected, since some scenarios were more ambiguous than others. The four answer choices presented for each question were designed by us and it was inevitable that some choices were more obvious to choose or to eliminate depending on the scenario and the question asked.

Furthermore, it is to be noted that all of our videos intentionally lacked obvious contextual clues. Lack of such clues is a usual situation when encountering a mobile robot like CoBot. Visitors often encounter the robot for the first time and interact with it with no knowledge at all about its capabilities, current state, or expectations from humans. Another example is when CoBot is stopped waiting for help (e.g., at an elevator). In such cases, looking at the robot from afar does not give much insight about what the robot's state, unless other visual cues are present.

Moreover, our three animations, originally designed for a specific scenario, are shown to generalize well to other similar scenarios. Some scenarios (like P3 and W3) even outperform the scenario used for the design of the light animations. We attribute the success of such generalizability to the abstraction used in these animations.



**Fig. 17** Comparison of the accuracies on each scenario across the two study conditions (error bars constructed using a 95% confidence interval; statistical significance shown used *t*-test). (Color figure online)



Furthermore, the fact that no significant changes in participants confidence was observed between the two conditions might be due to the fact that participants in one condition were unaware of the other condition which might have made all confidence reports average out around the mean score. It might be interesting to see if these results change in a within-subject study of the same type. The confidence people have on hypothesizing about a robot's state and actions is in fact an important factor to have in mind when designing robotic systems, especially safety-critical ones (such as for example self-driving cars).

Finally, it is important to note that the design of our study places the participants as a third person observing an interaction and not being part of it. Even though this design choice was made to isolate our measure of legibility, which could have otherwise been compromised by aspects such as motion, proximity, field of view, and so on, we could expect different results in a real world interaction, especially when it comes to trust. It would be interesting to investigate a more careful study of trust in a specifically designed task that would involve different animations, in the context of human–robot collaboration.

### 6.3 Experiment: Impact on Human Behavior

In order to evaluate the impact of the robot body lights on people's behavior with respect to the robot, we conducted a small observational study to evaluate whether the presence of lights influenced people to help CoBot at an elevator by pressing the elevator button. The experiment was run in the Computer Science (CS) building of our university, which is mainly populated by CS students and faculty, but also non-CS students, staff and visitors (exact participant background information was not gathered for this experiment).

#### 6.3.1 Experimental Procedure

CoBot was placed in front of the elevator with the following text on its screen: "Can you please press the elevator button for me to go up?". We had two experimental conditions:

- Condition "Lights on": in this condition, the lights were on and displaying our designed animation for the 'elevator waiting' scenario.
- Condition "Lights off": in this condition, the lights were off and the only indication that the robot needed help pressing the elevator button was the on-screen text.

The experimenter was sitting in the elevator area at a distance permitting accurate observation while not making the participants feel that they were being observed. The only information recorded was whether or not the person provided help to the robot. It was coded as a binary variable 'help', where '1' means help was provided and '0' means no help was provided.

The experiment was run over three different floors of our building (labeled GHC6, GHC7, and GHC8), which had slightly different elevator area architectures and different people. Both conditions were run over each floor, with the order of the conditions randomized and counterbalanced across floors.

#### 6.3.2 Data Filtering

Because the human dynamics around elevator areas can be complex, we had to filter our data according to some criteria, listed below:

- We filtered out people who passed across the robot from the back, since it would be hard for them to see what the screen said.

- If there was at least one person waiting at the elevator, any behavior from the passing people was discarded since the presence of people at the elevator might have interfered with people's willingness to help the robot (e.g., diffusion of responsibility effects).
- People who were not only passing across the elevator but planned to get on it were filtered out, unless they purposefully pressed the button in the reverse direction they were going to help the robot (i.e., pressing the 'up' button for the robot and the 'down' button for themselves).
- If the 'up' elevator button was already pressed and people showed a clear intent of pressing it again without actually pressing it, they were included.

According to our filtering criteria, we gathered a total of 10 data points for each floor/condition, resulting in a total of 60 participants.

### 6.3.3 Results and Discussion

Table 6 shows the results for the three floors. We observe that the presence of lights has a statistically significant impact on humans helping CoBot ( $t(58) = \pm 2.00$  two-tailed,  $p = 0.007$ ). We attribute this difference to two factors: first, the fact that the animation for this scenario is legible as shown in our legibility study; second, the fact that the animation was effective at grabbing people's *visual attention* in addition to the other contextual clues that the robot is in need of help such as the on-screen text and the position and orientation of the robot close to the elevator. While only 23.3% of people provided help to CoBot with lights off, 56.7% provided help when the lights were on. The effect was more or less pronounced according to the floor, which we may attribute to different factors such as familiarity with the robot, architectural differences between the floors that might affect visibility, and time of the day.

This study demonstrated in a single scenario that expressive lights, beyond legibility of expression (evaluated in user study 2), can have an actual impact on the behavior of humans around robots. In the elevator help scenario, expressive lights enabled better collaboration between the robot and the human, even when the humans who provided help were not the ones who requested the service from the robot (and hence, they are not getting anything in return from the robot by providing help to it). It would be interesting to see what behaviors are observed with/without expressive lights in a scenario where the human may get some benefit from not helping the robot (such as not wanting to move away from the path of the robot in order to play with it). Those results may or may not generalize, especially depending on the background of the person interacting, but a more careful investigation is definitely needed to hypothesize about those less simplistic scenarios.

**Table 6** Proportion of people who helped the robot in the two experimental conditions across the three floors

Floor	Lights on	Lights off
GHC7	7/10	1/10
GHC6	4/10	3/10
GHC8	6/10	3/10
Total (%)	56.7%	23.3%

## 6.4 Section Summary

In this section, we have presented three user studies which evaluate the behavior of CoBot's body light strip.

The first study informed our design of light animations for three different scenarios (waiting at an elevator, blocked by a human obstacle, and escorting a person). We asked experts to provide their top choice from a 'catalog' of available light animations, speeds, and colors.

The second study validated the legibility of the designed animations, as well as their generalizability to scenarios similar to the ones used in the design process. It mainly showed that the presence of lights has a significant impact on naive people's understanding of the robot in diverse situations.

The third study was a small real-world experiment which proved that these lights can have a real impact on people's behavior with regard to the robot. It showed that people were significantly more likely to provide help to a robot with expressive lights in a simple scenario.

## 7 Conclusion

This paper has focused on enabling robots to effectively communicate information about their state through the use of expressive lights. We focused on three main aspects of such communication modality in relation to robot state, summarized below.

First, we came up with **informative** expressive behaviors of the robot's state by: (1) *selecting* from it user-relevant elements as a function of situation and context, and (2) *mapping* them to the elements of our expression channel, namely expressive lights.

Second, we designed these expressive behaviors, namely, light animations, ensuring they are **legible**, such that they require minimal or no training to be understood by first-time users. We informed our design choices by both *design* and *evaluation* user studies.

Our technical contributions can be summarized as follows:

- An *efficient representation for robot state information* which builds upon *robot variables* and *state features* to

form structures suited for expression which we call *animation tuples*,

- A *formal framework for light animation control* which divides the animation space into a finite set of *signal shapes* with associated parameters related to *dynamics, color, intensity, and spatial indexing*. An *episodic animation control algorithm* is used to render animatable structures (*complete animation tuple sets*) into *animation frames* on a digital LED strip. Although the analysis is focused on addressable light strips, it easily generalizes to other forms of light arrays,
- A *design study* to investigate the design (parameter selection) of animating robot state through expressive lights, and
- Two *evaluation studies* which show that expressive lights on a robot can increase understanding of robot states and actions, but also have an influence on the behavior of humans interacting with the robot.

The formalism of both state representation and mapping as well as light animation control was made general enough to be easily utilized in different robots or technological devices, as well as different types of animated light sources. In this work, it was used to map the state of the mobile service robot CoBot to two light strips expressing different types of information about the robot's state. Our design and evaluation studies focused on a single light strip, with three light animations that were applied to three classes of scenarios. We are confident that other similar scenarios on mobile robots with different tasks and services, or even different types of robots, could benefit from using legible expressive lights in a way similar to what was done in this work, to enable a better understanding of the robot's operation, goals, and knowledge. Beyond legibility, even though we only touched on the effect of those lights on the idea of trust, it is of pressing importance to understand in the future how such a communication modality is able to positively or negatively affect more sophisticated measures of trust, but also collaboration, rapport, attachment, and more, in both short-term and long-term interactions.

Finally, the natural extension of this work is its application to truly multimodal robot expression. With multiple modalities, there is a problem of distributing expressive behaviors across different heterogeneous modalities with different capabilities and limitations. Also, investigating aspects of redundancy and complementarity will be important since modalities, relating to the same system, cannot be treated as completely independent. Along these lines, there is also the problem of synchronization between different modalities, especially when two modalities use the same physical resource (e.g., sound and speech, functional and expressive motion).

**Acknowledgements** This research was partially supported by the FCT INSIDE ERI grant, FLT Grant Number 2015-143894, NSF Grant Number IIS-1012733, and ONR Grant N00014-09-1-1031. The views and conclusions contained in this document are those of the authors only. The authors declare that they have no conflict of interest. We would like to thank Ana Paiva and Stephanie Rosenthal for their guidance on the user studies, as well Joydeep Biswas and Richard Wang for their development and maintenance of the autonomous CoBot robots.

## References

1. Alves-Oliveira P, Di Tullio E, Ribeiro T, Paiva A (2014) Meet me halfway: eye behaviour as an expression of robot's language. In: 2014 AAAI fall symposium series
2. Baraka K, Paiva A, Veloso M (2016) Expressive lights for revealing mobile service robot state. In: Robot 2015: second Iberian robotics conference. Springer, pp 107–119
3. Baraka K, Rosenthal S, Veloso M (2016) Enhancing human understanding of a mobile robot's state and actions using expressive lights. In: Robot and human interactive communication (RO-MAN), 2016 25th IEEE international symposium on. IEEE, pp 652–657
4. Bertin J (1983) Semiology of graphics. University of Wisconsin Press
5. Betella A, Inderbitzin M, Bernardet U, Verschure PF (2013) Non-anthropomorphic expression of affective states through parametrized abstract motifs. In: Affective Computing and Intelligent Interaction (ACII), 2013 Humaine association conference on. IEEE, pp 435–441
6. Bethel CL (2009) Robots without faces: non-verbal social human-robot interaction. Graduate theses and dissertations. <http://scholarcommons.usf.edu/etd/1855>
7. Choi Y, Kim J, Pan P, Jeung J (2007) The considerable elements of the emotion expression using lights in apparel types. In: Proceedings of the 4th international conference on mobile technology, applications, and systems. ACM, pp 662–666
8. De Lorenzo RA, Eilers MA (1991) Lights and siren: a review of emergency vehicle warning systems. Ann Emerg Med 20(12):1331–1335
9. De Melo C, Paiva A (2007) Expression of emotions in virtual humans using lights, shadows, composition and filters. In: Affective computing and intelligent interaction. Springer, pp 546–557
10. Dragan A (2015) Legible robot motion planning. Ph.D. Thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh
11. Funakoshi K, Kobayashi K, Nakano M, Yamada S, Kitamura Y, Tsujino H (2008) Smoothing human-robot speech interactions by using a blinking-light as subtle expression. In: Proceedings of the 10th international conference on multimodal interfaces. ACM, pp 293–296
12. Geratthewohl SJ (1957) Conspicuity of flashing light signals: effects of variation among frequency, duration, and contrast of the signals. J Opt Soc Am 47(1):27–29
13. Haddock SHD, Moline MA, Case JF (2010) Bioluminescence in the sea. Annu Rev Mar Sci 2(1):443–493. doi:10.1146/annurev-marine-120308-081028
14. Harrison C, Horstman J, Hsieh G, Hudson S (2012) Unlocking the expressivity of point lights. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, pp 1683–1692
15. Holmes K (2016) The mood of the chinese internet lights up the facade of beijing's water cube. <http://motherboard.vice.com/blog/video-the-great-mood-building-of-china> (n.d.). Accessed 11 Feb 2016
16. Hoonhout J, Jumpertz L, Mason J, Bergman T (2013) Exploration into lighting dynamics for the design of more pleasurable

- luminaires. In: Proceedings of the 6th international conference on designing pleasurable products and interfaces. ACM, pp 185–192
17. Jones DN (2016) Interactive light art show 'congregation' opens at market square. <http://www.post-gazette.com/local/city/2014/02/22/Interactive-light-art-show-opens-at-Pittsburghs-Market-Square/stories/201402220081> (2014). Accessed 11 Apr 2016
  18. Kim M, Lee, HS, Park JW, Jo SH, Chung MJ (2008) Determining color and blinking to support facial expression of a robot for conveying emotional intensity. In: Robot and Human interactive communication, 2008. RO-MAN 2008. The 17th IEEE international symposium on. IEEE, pp 219–224
  19. Knight H, Simmons R (2014) Expressive motion with  $x$ ,  $y$  and  $\theta$ : Laban effort features for mobile robots. In: Robot and Human interactive communication, 2014 RO-MAN: The 23rd IEEE international symposium on. IEEE, pp 267–273
  20. Kobayashi K, Funakoshi K, Yamada S, Nakano M, Komatsu T, Saito Y (2011) Blinking light patterns as artificial subtle expressions in human–robot speech interaction. In: RO-MAN, 2011 IEEE. IEEE, pp 181–186
  21. Langmuir I, Westendorp WF (1931) A study of light signals in aviation and navigation. *J Appl Phys* 1(5):273–317
  22. Lloyd JE (1971) Bioluminescent communication in insects. *Annu Rev Entomol* 16(1):97–122
  23. Mutlu B, Forlizzi J, Nourbakhsh I, Hodgins J (2006) The use of abstraction and motion in the design of social interfaces. In: Proceedings of the 6th conference on designing interactive systems. ACM, pp 251–260
  24. Feldmaier J, Marmat T, Kuhn J, Diepold K (2016) Evaluation of a RGB-LED-based emotion display for affective agents. *CoRR*. [arXiv:1612.07303](https://arxiv.org/abs/1612.07303)
  25. Perera V, Soetens R, Kollar T, Samadi M, Sun Y, Nardi D, van de Molengraft R, Veloso M (2015) Learning task knowledge from dialog and web access. *Robotics* 4(2):223–252
  26. Rea DJ, Young JE, Irani P (2012) The Roomba mood ring: an ambient-display robot. In: Proceedings of the seventh annual ACM/IEEE international conference on Human–Robot interaction. ACM, pp 217–218
  27. Rosenthal S, Biswas J, Veloso M (2010) An effective personal mobile robot agent through symbiotic Human–Robot interaction. In: Proceedings of AAMAS'10, the ninth international joint conference on autonomous agents and multi-agent systems. Toronto
  28. Schanda J (2007) Colorimetry: understanding the CIE system. Wiley, Hoboken
  29. Seitinger S, Taub DM, Taylor AS (2010) Light bodies: exploring interactions with responsive lights. In: Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction. ACM, pp 113–120
  30. Song S, Yamada S (2017) Expressing emotions through color, sound, and vibration with an appearance-constrained social robot. In: Proceedings of the 2017 ACM/IEEE international conference on human–robot interaction, HRI '17. ACM, New York, pp 2–11
  31. Stricker R, Miller S, Einhorn E, Schrter C, Volkhardt M, Debes K, Gross HM (2012) Interactive mobile robots guiding visitors in a university building. In: RO-MAN. IEEE, pp 695–700
  32. Szafrir D, Mutlu B, Fong T (2015) Communicating directionality in flying robots. In: Proceedings of the tenth annual ACM/IEEE international conference on human–robot interaction. ACM, pp 19–26 (2015)
  33. Veloso M, Biswas J, Coltin B, Rosenthal S (2015) CoBots: robust symbiotic autonomous mobile service robots. In: Proceedings of IJCAI'15, the international joint conference on artificial intelligence. Buenos Aires
  34. Wolfe JM, Horowitz TS (2004) What attributes guide the deployment of visual attention and how do they do it? *Nat Rev Neurosci* 5(6):495–501
  35. Wright A (2009) The colour affects system of colour psychology. In: AIC quadrennial congress, 2009
  36. Wright B, Rainwater L (1962) The meanings of color. *J Gen Psychol* 67(1):89–99
  37. Xia G, Tay J, Dannenberg R, Veloso M (2012) Autonomous robot dancing driven by beats and emotions of music. In: Proceedings of the 11th international conference on autonomous agents and multiagent systems-volume 1, pp 205–212

**Kim Baraka** is currently a dual degree Ph.D. student in Robotics at Carnegie Mellon's Robotics Institute (Pittsburgh, PA, USA), and Instituto Superior Técnico (Lisbon, Portugal). He holds an M.S. in Robotics from Carnegie Mellon, and a Bachelor in Electrical and Computer Engineering from the American University of Beirut. His research interests lie at the intersection between Artificial Intelligence, Machine Learning and Human-Robot Interaction. A believer in the coexistence of robots and humans, his graduate research focuses on making robots both more adaptive and more transparent to humans. More specifically, he is recently interested in the development of adaptive algorithms for socially assistive robotics, such as the use of robots in autism therapy.

**Manuela M. Veloso** is the Herbert A. Simon University Professor in the School of Computer Science at Carnegie Mellon University. She is the Head of the Machine Learning Department, with joint appointments in the Computer Science Department, in the Robotics Institute, and in the Electrical and Computer Engineering Department. She researches in Artificial Intelligence with focus in robotics, machine learning, and multiagent systems. She founded and directs the CORAL research laboratory, for the study of autonomous agents that Collaborate, Observe, Reason, Act, and Learn, [www.cs.cmu.edu/~coral](http://www.cs.cmu.edu/~coral). Professor Veloso is ACM Fellow, IEEE Fellow, AAAS Fellow, AAAI Fellow, Einstein Chair Professor, the co-founder and past President of RoboCup, and past President of AAAI. Professor Veloso and her students research with a variety of autonomous robots, including mobile service robots and soccer robots. See [www.cs.cmu.edu/~mmv](http://www.cs.cmu.edu/~mmv) for further information, including publications.