A *High-Resolution Intelligence* Implementation based on *Design-to-Robotic-Production and -Operation* strategies

A. Liu Cheng[a], H. H. Bier[a], G. Latorre[b], B. Kemper[a], and D. Fischer[a]

[a]Faculty of Architecture and the Built Environment, Delft University of Technology, Delft, The Netherlands
[b]Facultad de Ingeniería de Sistemas, Escuela Politécnica Nacional, Quito, Ecuador
E-mail: a.liucheng@tudelft.nl, h.h.bier@tudelft.nl, galoget.latorre@epn.edu.ec, b.n.kemper@student.tudelft.nl, d.l.fischer@student.tudelft.nl

Abstract –

This paper presents an initial *proof-of-concept* implementation of a comprehensively intelligent built-environment based on mutually informing *Design-to-Robotic-Production and -Operation* (D2RP&O) strategies and methods developed at *Delft University of Technology* (TUD). In this implementation, D2RP is expressed via deliberately differentiated and function-specialized components, while D2RO expressions subsume an extended *Ambient Intelligence* (AmI) enabled by a *Cyber-Physical System* (CPS). This CPS, in turn, is built on a heterogeneous, scalable, self-healing, and partially meshed *Wireless Sensor and Actuator Network* (WSAN) whose nodes may be clustered dynamically *ad hoc* to respond to varying computational needs.

Two principal and innovative functionalities are demonstrated in this implementation: (1) cost-effective yet robust *Human Activity Recognition* (HAR) via *Support Vector Machine* (SVM) and k-*Nearest Neighbor* (*k*-NN) classification models, and (2) appropriate corresponding reactions that promote the occupant's spatial experience and well-being via continuous regulation of illumination with respect to colors and intensities to correspond to engaged activities.

The present implementation attempts to provide a fundamentally different approach to intelligent built-environments, and to promote a highly sophisticated alternative to existing intelligent solutions whose disconnection between architectural considerations and computational services limits their operational scope and impact.

Keywords –

Design-to-Robotic-Production and -Operation, Cyber-Physical Systems, Adaptive Architecture, Wireless Sensor Networks, Ambient Intelligence.

1 Introduction

The present paper promotes *Design-to-Robotic-Production and -Operation* (D2RP&O) [1] strategies and methods as drivers of highly sophisticated *Ambient Intelligence* (AmI) solutions, and demonstrates its competence in this endeavor by presenting and describing a corresponding high-resolution intelligence implementation. Two principal and innovative functionalities are described in this implementation, the first pertaining to computational intelligence while the second to architectural variables / considerations (see Section 2).

With respect to the first functionality, a *Machine Learning* (ML) subsystem is integrated in the proposed system-architecture in order to enable *Human Activity Recognition* (HAR) mechanisms. With respect to HAL, ML methods have typically used gyroscopic data collected via portable devices (e.g., smartphones, etc.) [2, 3] or via sensor-fusion [4]. The ML subsystem consists of two classification mechanisms developed based on polynomial programming of *Support Vector Machine* (SVM) and k-*Nearest Neighbor* (*k*-NN) classifiers. These SVM and *k*-NN models are built on a dynamically clustered set of high-performance nodes in the localized *Wireless Sensor and Actuator Network* (WSAN). Cloud-based SVM and *k*-NN counterparts are generated as alternatives to the localized mechanisms for contingency measures (see Section 3.1). With respect to the second functionality, an interactive / adaptive illumination system capable of identifying and mitigating—via said ML mechanisms—fatigue via regulation of colors and corresponding intensities is proposed (see Section 3.2).

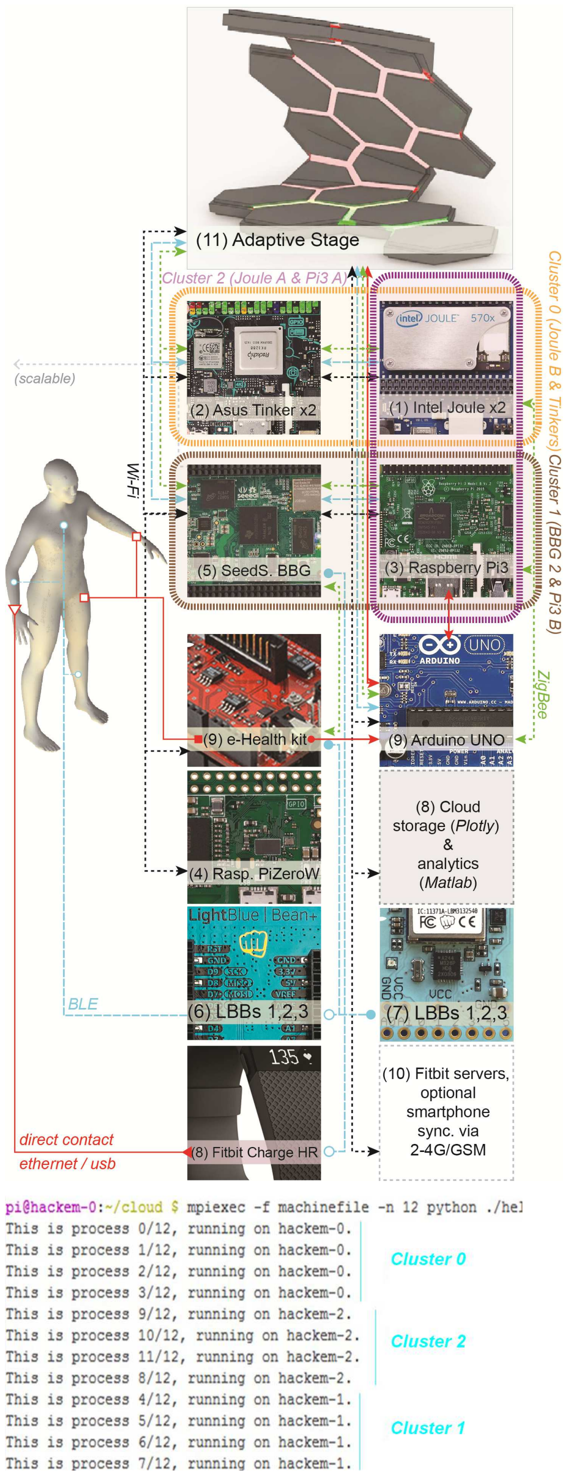
The comprehensive character of the intelligence imbued in said implementation supervenes on the mutually corresponding and informing relationship between computational mechanisms and architectural considerations.

2 Concept and Approach

The present implementation continues to build on the adaptive mechanisms and system-architecture previously outlined and developed by the authors [5–7] (see Figure 1). With respect to intelligence in the built-environment, it revisits *Protospace 4.0*'s [8–10] system of function-specific differentiated components. On the occasion of the international conference *Game Set and Match 3* (GSM3) [11] held at the *Faculty of Architecture and the Built-Environment, Delft University of Technology* (TUD) (9th-11th of November, 2016), a fragment of *Protospace 4.0* was rebuilt as a responsive stage, and a purpose-built interactive LED-based illumination system was integrated into its architecture (see Figure 5). This illumination system serves as a subsystem of the present system-architecture.

With respect to computational intelligence, an ML framework is deployed as a subsystem to enable cost-effective yet robust HAR mechanisms via established classifications models—i.e., SVM and *k*-NN. A smartphone as well as three *Light Blue Beans*™ (LBBs) were used to gather gyroscopic and accelerometer data from the user via the *Open Sound Control* (OSC) protocol. The generated dataset was used to train two SVM and *k*-NN models, one via local clusters using open-source and purpose-written *Python* scripts, and another via an external computer—simulating cloud-based analytics services—using third-party proprietary software. The principal intention was to imbue the proposed system with both localized as well as web-based analysis mechanisms in order to ascertain ML robustness and resilience in case either mechanism failed. A secondary intention was (a) to demonstrate that open-source solutions could be as effective as those rendered by proprietary software while reducing costs; and (b) to illustrate how purpose-written scripts integrated more seamlessly and efficiently (in terms of interoperability) than did proprietary software.

The integration of both built-environment as well as computational intelligences instantiates a *high-resolution intelligence* environment capable of translating sensed data into informed and correlated active, reactive, and interactive responses pertinent to the activities engaged by the user. The SVM and *k*-NN mechanisms are trained to identify certain data values as corresponding to a variety of activities, and to use this prediction power to dynamically mitigate fatigue in the user via an active and adaptive regulation of colors and intensities (see Section 3.2.2). Moreover, the responsive stage is also imbued with predetermined behavioral patterns such as pulsating when idle, tracing paths, correlating different colors to identified body parts (via Microsoft® *Kinect*™ V2) of up to six different individuals (see Section 3.2.1).



```
pi@hackem-0:~/cloud $ mpiexec -f machinefile -n 12 python ./hel
This is process 0/12, running on hackem-0.
This is process 1/12, running on hackem-0.
This is process 2/12, running on hackem-0.
This is process 3/12, running on hackem-0.
This is process 9/12, running on hackem-2.
This is process 10/12, running on hackem-2.
This is process 11/12, running on hackem-2.
This is process 8/12, running on hackem-2.
This is process 4/12, running on hackem-1.
This is process 5/12, running on hackem-1.
This is process 6/12, running on hackem-1.
This is process 7/12, running on hackem-1.
```

Figure 1: Top: Heterogeneous system-architecture with three dynamic *ad hoc* clusters. Bottom: Runtime processes distribution.

3 Methodology and Implementation

The development of the detailed implementation consists of three parts: (1) the design and development of cost-effective HAR system (see Section 3.1), which involved the development of (1a) a dynamic *ad hoc* heterogeneous clustering system (see Section 3.1.1) as well as (1b) data-gathering and -parsing scripts for ML training and testing purposes (see Section 3.1.2); (2) the design and installation of the LED-based illumination subsystem and its corresponding electronic setup (see Section 3.2); (3) the integration of the previous parts into a unified closed-loop system architecture. The first and second parts were developed in parallel and tested as working subsystems before integration.

3.1 Development of a cost-effective *Human Activity Recognition* (HAR) system

Due to their evolving and resilient characters, ML classifiers have been implemented in a variety of applications built on WSANs [12]. HAR, as one such application, has successfully exploited said classifiers in the last five years (see, for example, [13–15]). However, due to the cost-effective and low energy-consumption character typical of WSAN nodes, computational processing with respect to feature extraction has been considerably limited [16]. To overcome this limitation, the present implementation is capable of instantiating *ad hoc* clusters consisting of a variety of high-performance nodes. Furthermore, several clusters may be instantiated simultaneously in order to enable parallel high-performance information processing activities.

Another way to overcome this limitation is to avoid it altogether by outsourcing all high-performance information processing to cloud-based ML services (e.g., Google® *CloudPlatform*™, Amazon® *Machine Learning*™, Microsoft® *Azure*™, etc.). But there are a number of limitations with this approach. The first, and perhaps the most salient, is the cost incurred by including proprietary services in any proposed intelligent built-environment solution. A second yet no less important limitation may be the impact to the solution's resilience. That is to say, should said built-environment lose access to the Internet, it would be incapable of generating classification models.

The present implementation proposes the integration of both cloud-based as well as localized ML capabilities in order to ascertain robustness and resilience. Whenever possible, ML processes are locally and dynamically executed via *ad hoc* node-clustering. But should this prove impossible either due to failure or unavailability of proper resources, cloud-based ML services are used.

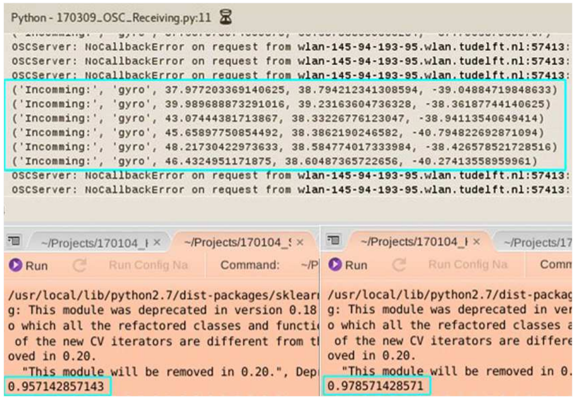
3.1.1 Dynamic Clustering mechanism

The system's clustering mechanism uses the *Message Passing Interface* (MPI) standard via *MPI for Python* (mpi4py) [17] (see Figure 1, *Bottom*). The system's ecosystem consists of nine types of development platforms, *Microcontroller Units* (MCUs), and proprietary trackers: (1) Intel® *Joule*™, (2) Asus® *Tinkerboard*™, (3) Raspberry® *Pi 3*™ and (4) *Pi Zero W*™, (5) SeedStudio® *BeagleBone Green*™ (BBG), (6) Punch Through® *Bean+*™ and (7) LBB, (8) Fitbit® *Charge HR*™, and (9) Arduino® *UNO*™ (see Figure 1). Sets of items 1, 2, 3, and 5 may be dynamically clustered *ad hoc* via WiFi for high-performance information processing, and are connected to the rest of the network via WiFi, ZigBee, BLE wireless communication protocols and—in the case of an instance of item 3—Ethernet / USB cables. Items 4, 7, and 9 are considered as low-computation *end devices* meshed into the WSAN via ZigBee, with 6 serving as router for 7 via BLE. Since there is a direct relationship between computational power vs. energy-consumption, *end device* and *router* nodes are concerned exclusively with sensor-data gathering and relaying with minimal information processing. Depending on the task, nodes exchange data via pertinent protocols and frequencies.

3.1.2 *Machine Learning* (ML) mechanisms

As detailed in Section 2 and Section 3.1, two ML mechanisms are integrated into the present implementation: (1) a localized *ad hoc* cluster system based on open-source and purpose-written *Python* scripts, and (2) a simulated cloud-based analytics service using MathWorks® *MATLAB*™. In both mechanisms SVM and *k*-NN classification models are generated.

In the localized mechanism, a script based on *pyOSC* is first written to receive OSC data from any device and application capable of broadcasting in said protocol. While all the WiFi-enabled nodes in the system's WSAN have the capacity to receive this data-streaming, only one of the nodes of the cluster instantiated to generate classification models stores it locally and streams it to a cloud-based data visualization service (i.e., *Plotly*™). Should the receiving node fail, another high-performance node will replace it automatically. Since the proposed solution uses a smartphone and three LBBs for data redundancy, resolution, and validation, the script in question proceeds to parse and to reduce the noise in the received multi-sensor data in order to generate a robust and unified dataset. At this point the dataset is processed through two ML scripts based on *scikit-learn* [18, 19], one for SVM and another for *k*-NN classification models (see Figure 2).



```
OSCServer: NoCallbackError on request from wlan-145-94-193-95.wlan.tudelft.nl:57413:
OSCServer: NoCallbackError on request from wlan-145-94-193-95.wlan.tudelft.nl:57413:
OSCServer: NoCallbackError on request from wlan-145-94-193-95.wlan.tudelft.nl:57413:
('Incomming:', 'gyro', 37.977203369140625, 38.794212341308594, -39.04884719848633)
('Incomming:', 'gyro', 39.989688873291016, 39.23163604736328, -38.36187744140625)
('Incomming:', 'gyro', 43.07444381713867, 38.33226776123047, -38.94113540649414)
('Incomming:', 'gyro', 45.65897750854492, 38.3862190246582, -40.794822692871094)
('Incomming:', 'gyro', 48.21730422973633, 38.584774017333984, -38.426578521728516)
('Incomming:', 'gyro', 46.4324951171875, 38.60487365722656, -40.27413558959961)
OSCServer: NoCallbackError on request from wlan-145-94-193-95.wlan.tudelft.nl:57413:
OSCServer: NoCallbackError on request from wlan-145-94-193-95.wlan.tudelft.nl:57413:
```

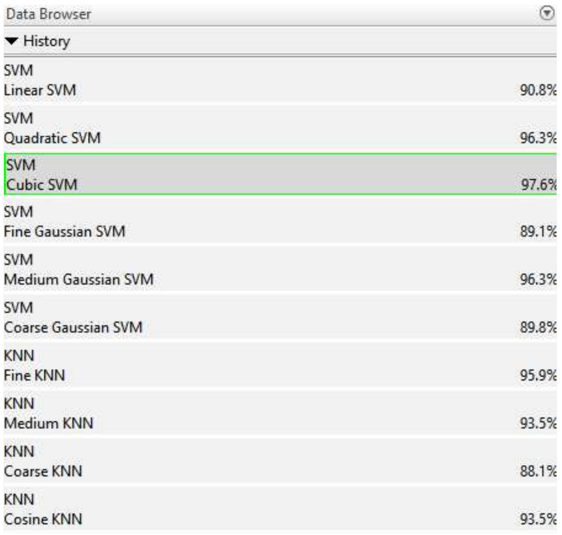
```
/usr/local/lib/python2.7/dist-packages/sklear
g: This module was deprecated in version 0.18
o which all the refactored classes and functi
 of the new CV iterators are different from t
oved in 0.20.
  "This module will be removed in 0.20.", Dep
0.957142857143
```

```
/usr/local/lib/python2.7/dist-packag
g: This module was deprecated in ver
o which all the refactored classes a
 of the new CV iterators are differe
oved in 0.20.
  "This module will be removed in 0.
0.978571428571
```

Figure 2: Top: OSC-data receiving and parsing. Bottom: 95.7% prediction success with respect to HAR via SVM (left) vs. 97.85% via *k*-NN (right).

It should be noted that each time a classification model is generated, regardless of whether it is done via open-source or proprietary means, its resulting prediction success rate will vary. For the purposes of the present discussion, the success rate generated in the last sample run is used. That is to say, the success rate of the localized SVM mechanism was 95.7% while that of the *k*-NN mechanism 97.85%.

In the proprietary cloud-based mechanism, as simulated by a computer external to the system's WSAN and running MATLAB™, the same datasets are processed through several *Classification Learners* (see Figure 3).



SVM Linear SVM	90.8%
SVM Quadratic SVM	96.3%
SVM Cubic SVM	97.6%
SVM Fine Gaussian SVM	89.1%
SVM Medium Gaussian SVM	96.3%
SVM Coarse Gaussian SVM	89.8%
KNN Fine KNN	95.9%
KNN Medium KNN	93.5%
KNN Coarse KNN	88.1%
KNN Cosine KNN	93.5%

Figure 3: Sample MATLAB™-generated ML models with corresponding success rate.

It may be observed that the most successful classification model generated by MATLAB™ is based on *cubic* SVM with a prediction success rate of 97.6%, which is higher than the rate corresponding to the localized and open-source SVM result (i.e., 95.7%). But it may also be noted that the localized and open-source *k*-NN success rate is higher than any of the *k*-NN models generated by MATLAB™ (i.e., 97.8% vs. 95.9%, 93.5%, 88.1%, and 93.5%). It may not be inferred from this that the localized is superior to the proprietary, nor vice versa. It may be considered, however, that the localized and open-source mechanism yields comparably robust results as that of the proprietary one within the scope of the present implementation.

Having generated two sets of classification models via localized and cloud-based means, the ones with the most successful prediction rate are used at runtime, with precedence given to the localized mechanism—if and only if said mechanism fails or has unavailable resources are cloud-based ML models be used. The duration of said runtime may be determined by the user, but it should be as brief as practicable in order for the dataset to be updated with new data. For example, the user may decide to schedule the generation of a new updated model every 24 hours and only during sleep periods. This way the user would wake up to updated and relatively more attuned models every day. Furthermore, via this incrementally updating process, classification models may be trained to detect and/or predict new activities or patterns in a gradual manner, thereby enabling the intelligent built-environment to evolve with its user.

3.2 Development of the architecture-embedded interactive / adaptive illumination subsystem

As detailed in Section 2, a fragment of *Protospace 4.0* was repurposed to conform a responsive stage on the occasion of the *GSM3* conference. 16 differentiated and function-specific components, viz., *protoCELLs* [10], are assembled to conform said stage while integrating a custom-designed and -built interactive / adaptive LED-based illumination system. The indented borders of each component were lined with LED-strips, which enabled individual color control. In conjunction, these indented borders create a continuous indented seam between all components, which is covered with translucent material in order to enable diffusion of color and intensity. The combination between this translucent cover with two separate individually controlled LED-strips enables the instantiation of multiple color gradients and intensities (see Figure 4).

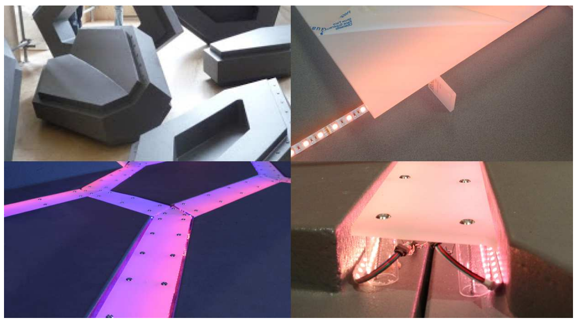


Figure 4: Top: Single *protoCELLs* (Left); testing *Perspex*® over LED1 (Right). Bottom: Generated color gradient (Left); Acrylic connections & two LED-strips within the seam of two *protoCELLs* (Right).

The system-architecture of the interactive / adaptive illumination system involves 12 Arduino® UNO™ MCUs that are physically connected to a computer via USB hubs. In the implemented revision, this computer is replaced by an Intel® *Joule*™ (see Figure 1), thereby integrating the stage and its responsive illumination into the system-architecture ecosystem of the present implementation. As a stand-alone system, the stage is configured to behave in particular and predetermined patterns. As a subsystem of a more sophisticated system-architecture, it is now imbued with ML capabilities for non-predetermined actions, reactions, and interactions.

3.2.1 Predetermined Scenarios

There are three predetermined scenarios: (1) Pulsating, (2) Lecture, and (3) Break, all of which are described as follows:

In the first scenario, as soon as the illumination system is powered, the stage slowly pulsates in one color—i.e., oscillates between intensities of a same color. This creates an effect viscerally reminiscent of a beating heart, tacitly suggesting that the stage is "alive". The intention of this scenario is to instigate interest and curiosity in the users, inviting them to engage with it (see Figure 5, *Top image 2*).

In the second scenario, two different types of interaction are envisioned during the conference presentations. The first involves the stage's reaction towards the movements of the speaker, where by stepping on or touching one or multiple components he/she instigates a gradual shift from the initial or *passive* colors to *active* colors for a certain period of time, after which *active* colors would default back to *passive* ones (see Figure 5, *Top images 3, 4*). The second interaction inverts this causal relationship to have the stage influence the speaker—i.e., the speaker knows his/her time is up when the first type of interaction ceases and the stage defaults back to a single color (see Figure 5, *Top image 5*).

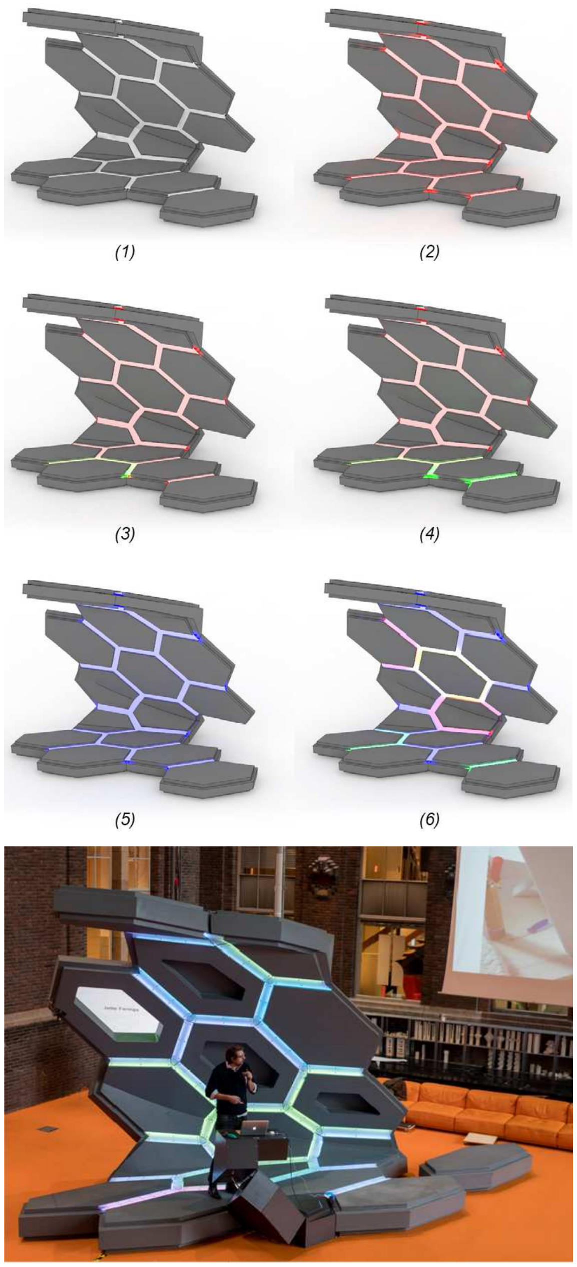


Figure 5: Top: images (1) 3D model with lights off; (2) Pulsating; (3) Activation of discrete components; (4) Leaving a trace; (5) Manual override; (6) individual activity detection correlated with color. Bottom: Implemented fragment.

In the third scenario, the stage invites interaction from the audience in-between lectures by allowing them to "paint the stage" via body gestures. That is, in this mode, the stage tracks body parts of up to six individuals and instantiates corresponding color changes across the components, hence correlating certain movements or body parts with certain colors (see Figure 5, *Top image 6*).

Finally, it should be noted that in addition to these three automated *cause-and-effect* scenarios, the illumination system is also designed with a manual override control. A proprietary fee-based Apple®'s iOS™ *application*, viz., *TouchOSC*™ (by Hexler Limited®) is used to develop customized control screens to provide override capabilities to the illumination system (see Figure 6).

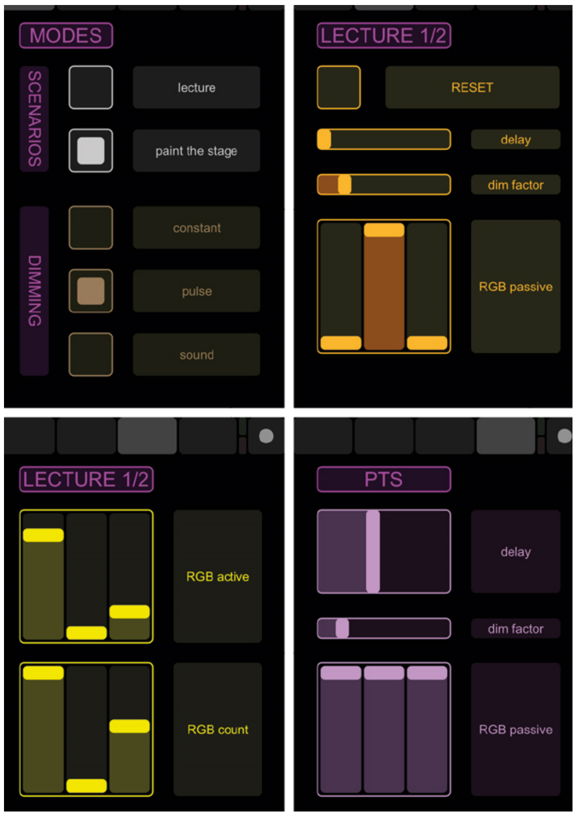


Figure 6: iOS OSC Applications: *TouchOSC* (proprietary).

3.2.2 Non-predetermined Scenario

The ML-driven HAR mechanism implemented in the present system (see Section 3.1.2), in conjunction with an adaptation of the human state estimation mechanism developed by Nakaso *et al.* [20], is used to detect general fatigue in the user. By learning from the user's behavior as a consequence of lighting conditions—both in terms of colors and intensities—the system can learn to identify which combinations of colors and intensities ameliorate or exacerbate the user's fatigue. Having made this identification, the illumination system continuously seeks to improve the state of the user by regulating the experience of the ambiance. Unlike predetermined scenarios, the system is not programmed to associate a given color with a given human state or action—nor vice versa—but rather the ML mechanisms establish such correlations as processed via HAR. More specifically, the localized *k*-NN classification model is capable of learning to predict which colors and intensities are conducive to mood amelioration / fatigue mitigation and to promote them. Such colors and intensities may change over time, as saturations in the frequency of particular colors and intensities over short periods of time could actually instigate an adverse effect. The ML mechanisms, however, can account for this change as they evolve accordingly.

Like the OSC-enabled manual override provided in the predetermined scenarios, the present scenario also integrates a *correction* mechanism based on human intervention. However, unlike the manual override, the *correction* mechanism is used to provide feedback—i.e., to "teach"—the system when a prediction is inaccurate. This fact is then considered in the next iteration of a new and updated classification model. The *correction* mechanism is implemented via a free and open-source OSC iOS™ *application*, viz., *Control* (by Charlie Roberts) (see Figure 7).

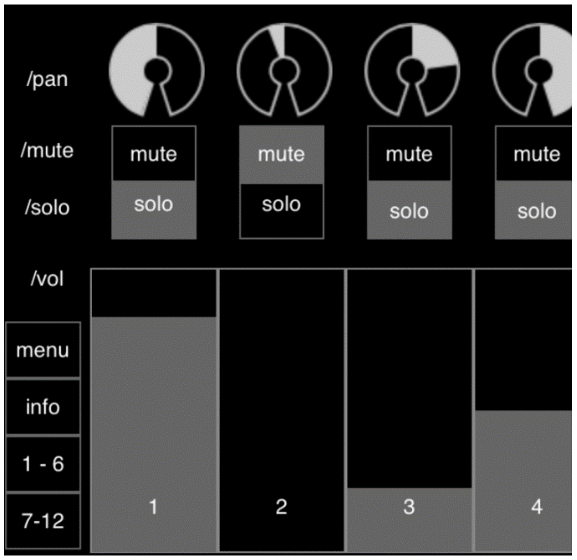


Figure 7: iOS OSC Applications: *Control* (open-source).

4 Conclusions and Future Work

The present paper attempts to promote a highly sophisticated intelligent built-environment framework based on D2RP&O principles and methodologies. It does so by presenting an implementation where sophisticated intelligence is imbued both in physical as well as computational terms. It promotes high-resolution computational intelligence by integrating ML mechanisms for HAR via a subsystem of dynamic *ad hoc* clustering. It also promotes high-resolution intelligence in terms of the built-environment by demonstrating how an interactive / adaptive illumination system can learn—via the detailed ML mechanisms—to reduce user-fatigue via the promotion of certain colors and intensities and the mitigation of others. Finally, by integrating both kinds of intelligences, the present implementation demonstrates the feasibility of a more sophisticated heterogeneous, open and scalable, and open-source AmI solution. Nevertheless, further work must be conducted in order to validate said feasibility in more complex scenarios, with more explicit and sophisticated expressions of D2RP for the integration of intelligence in the built-environment, and with correspondingly sophisticated expressions of D2RO with respect both to advances in ICTs as well as to ML mechanisms based on *Artificial Neural Networks* (ANNs).

A next iteration of a high-resolution intelligence implementation is presently being developed (1) to implement ANN-based ML; (2) to enhance fatigue detection; and (3) to extend the built-environment's ICT ecosystem to be open to more proprietary products and protocols.

With respect to the first task: in the present responsive stage implementation, the system detects fatigue exclusively via the analysis of detected patterns in the eyes (i.e., eyelids) of the user in conjunction with HAR data. However, due to the ambiguous character of human activity, it is difficult to differentiate a fatigued gait from a slow yet healthy one. The fatigue-detection precision may be improved by building on Ogawa *et al.*'s [21] facial-recognition and visual knee-position detection methods.

With respect to the second task: supervised and unsupervised learning ANN models are being explored for their suitability to given tasks. It is not necessarily the case that ANN models must be superior to SVM or *k*-NN models, as performance depends on the complexity, scale, and scope of the given tasks. At present a comparison matrix of ML models is being developed.

With respect to the third task: the system-architecture that the authors have been developing incrementally is highly heterogeneous both in terms of hardware, software, and communication protocols. This will continue to be a core interest in subsequent work. For example, present work is being conducted (A) to integrate Amazon®'s *Alexa Voice Service*™ into the ecosystem of the intelligent built-environment; and (B) to implement LoRaWAN as a *Low Power Wide Area Network* (LPWAN) protocol in order to extend interactions between high-resolution intelligence built-environments, whether these be private or public.

Acknowledgements

This paper has profited from the contribution of *Hyperbody* researchers and students involved in the described project.

References

[1] H. H. Bier, "Robotic Building as Integration of Design-to-Robotic-Production & Operation," *Next Generation Building*, no. 3, 2016.

[2] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A Public Domain Dataset for Human Activity Recognition Using Smartphones," in *Proceedings of the 21th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN 2013,* 2013.

[3] J. L. R. Ortiz, *Smartphone-based human activity recognition*. Cham: Springer, 2015.

[4] F. Palumbo, C. Gallicchio, R. Pucci, and A. Micheli, "Human activity recognition using multisensor data fusion based on Reservoir Computing," *AIS*, vol. 8, no. 2, pp. 87–107, 2016.

[5] A. Liu Cheng and H. H. Bier, "An Extended Ambient Intelligence Implementation for Enhanced Human-Space Interaction," in *Proceedings of the 33rd International Symposium on Automation and Robotics in Construction (ISARC 2016),* 2016.

[6] ____, "Adaptive Building-Skin Components as Context-Aware Nodes in an Extended Cyber-Physical Network," in *Proceedings of the 3rd IEEE World Forum on Internet of Things*: IEEE, 2016, pp. 257–262.

[7] A. Liu Cheng, C. Georgoulas, and T. Bock, "Fall Detection and Intervention based on Wireless Sensor Network Technologies," *Automation in Construction*, 2016.

[8] H. H. Bier, "Robotic building(s)," *Next Generation Building*, no. 1, 2014.

[9] H. H. Bier and S. Mostafavi, "Robotic Building as Physically Built Robotic Environments and Robotically Supported Building Processes," in *Human-computer interaction series, Architecture and interaction: Human computer interaction in*

space and place, N. S. Dalton, H. Schnädelbach, M. Wiberg, and T. Varoudis, Eds, Switzerland: Springer International Publishing, 2016, pp. 253–271.

[10] K. Oosterhuis, *004 | ProtoCELL*. Available: http://www.oosterhuis.nl/?p=23 (2017, Mar. 29).

[11] Hyperbody®, *GSM3: Game Set and Match: International Symposium & Exhibition.* Available: http://gsm3.hyperbody.nl/ (2016, Nov. 09).

[12] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, “Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications,” *IEEE Commun. Surv. Tutorials*, vol. 16, no. 4, pp. 1996–2018, 2014.

[13] W. Xiao and Y. Lu, “Daily Human Physical Activity Recognition Based on Kernel Discriminant Analysis and Extreme Learning Machine,” *Mathematical Problems in Engineering*, vol. 2015, no. 1, pp. 1–8, 2015.

[14] A. E. Villa, *Artificial neural networks and machine learning-- ICANN 2012: 22nd International Conference on Artificial Neural Networks, Lausanne, Switzerland, September 11-14, 2012, Proceedings*. Berlin, New York: Springer, 2012.

[15] J. Andreu and P. Angelov, “An evolving machine learning method for human activity recognition systems,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 4, no. 2, pp. 195–206, https://link-springer-com.ezproxy.library.ubc.ca/content/pdf/10.1007%2Fs12652-011-0068-9.pdf, 2013.

[16] E. L. Salomons, P. J. M. Havinga, and H. van Leeuwen, “Inferring Human Activity Recognition with Ambient Sound on Wireless Sensor Nodes,” (ENG), *Sensors (Basel, Switzerland)*, vol. 16, no. 10, 2016.

[17] L. D. Dalcin, R. R. Paz, P. A. Kler, and A. Cosimo, “Parallel distributed computing using Python,” *Advances in Water Resources*, vol. 34, no. 9, pp. 1124–1139, 2011.

[18] F. Pedregosa, and G. Varoquaux et al, *Scikit-learn: Machine Learning in Python.* Available: http://arxiv.org/pdf/1201.0490.

[19] L. Buitinck, and G. Louppe et al, *API design for machine learning software: experiences from the scikit-learn project.* Available: http://arxiv.org/pdf/1309.0238.

[20] S. Nakaso, J. Güttler, A. Mita, and T. Bock, “Human state estimation system implemented in an Office Deployable Getaway based on multiple bio information,” in *Proceedings of the 33rd International Symposium on Automation and Robotics in Construction (ISARC 2016)*, 2016.

[21] A. Ogawa, A. Mita, C. Georgoulas, and T. Bock, “A Face Recognition System for Automated Door Opening with parallel Health Status Validation Using the Kinect v2,” in *Proceedings of the 33rd International Symposium on Automation and Robotics in Construction (ISARC 2016)*, 2016, pp. 132–140.