Full/Regular Research Paper. Submission to CSCI-RTAI

Autonomous Sprinkler System with MAPE-K

Mira Kim
IBM Automation
International Business Machines Corporation[1]
Costa Mesa, CA, 92626
mirakim1012@gmail.com

Jennifer Jin[2]
Computer Science and Engineering
California State University
San Bernandio, CA 92407, USA
jennifer.jin@csusb.edu

***Abstract*—Sprinkler is a key component for farms and gardens. However, conventional sprinkler systems have limitations: underwatering and overwatering under evolving weather, not considering plant conditions, and burden of manual scheduling. We developed Autonomous Sprinkler System to remedy the limitations. We acquire environmental contexts with IoT sensors, cameras, and actuators, and apply MAPE-K for autonomic control. We implemented the Knowledge with a SVM classifier for inferring the situations and a Fast R-CNN model for determining plant condition. Our experiments show the system yields a considerable savings of water consumption and promotes the plant health, all without human users' interventions.**

***Keywords*—Smart Farm/Garden, Sprinkler System, Autonomic Computing, MAPE-K, Machine Learning**

I. Introduction

Smart farm or garden is a systems approach to fostering healthiness of plants and soil with an infrastructure to leverage advanced technology: IoT sensors, actuator devices, sensor fusion, data analytics, and actuation [1]. They together provide the capability of tracking, monitoring, analyzing, and automating farming or gardening operations [2].

A key component of smart farm/garden is *Sprinkler*, which is a hardware device used to spray water on plants or grass. A sprinkler system is an integrated set of devices and a software controller to provide a cost-efficient watering capability as shown in Fig. 1.

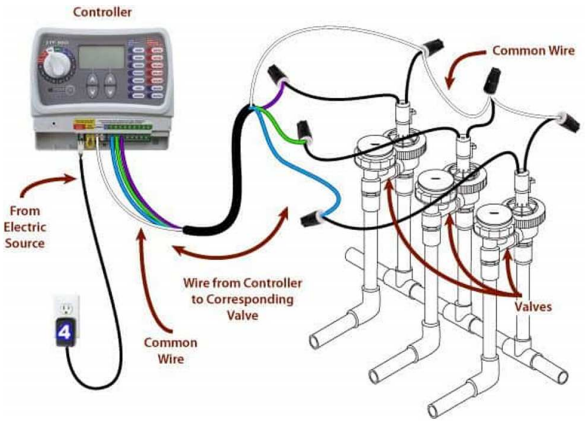


Fig. 1. Elements of Sprinkler System

Meanwhile, autonomic computing is the system capability that can manage itself without the intervention of human users through adaptive technologies. It is characterized by self-* capability as specified in MAPE-K model, consisting of four phases [3].

Monitor phase is to acquire environmental contexts using sensors and IoT devices. *Analyze phase* is to determine the environmental situation using acquired contexts. *Plan phase* is to devise an effective remedy action plan for the determined situation. *Execution phase* is to self-execute the devised action plan, and then the environment would be changed. The core of MAPE-K is the *Knowledge* that is used to govern the four operations.

The goal of our research is to devise a software framework that can autonomously manage Sprinkler operations for maintaining healthier plants and soil, by using MAPE-K model as shown in Fig. 2.

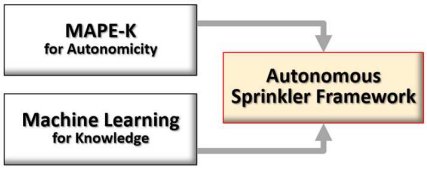


Fig. 2. Underlying Paradigms applied to Autonomous Sprinkler System

For the design of the framework, we applied object-oriented modeling, design patterns, architecture styles, machine learning classification and deep learning models, and closed-loop design for self-* controllability.

The key benefits of autonomous sprinkler system are (1) healthier plants and soil, (2) reduced water consumption by Sprinklers, (3) free of human users' intervention.

In this paper, we present the design and implementation of the Autonomous Sprinkler framework using machine learning algorithms. And we present a series of experiments on a single-family house in California for 12 months. We believe that the presented framework provides a foundation for building various types of smart farm and garden control systems.

II. Related Works

Most of works on MAPE-K are to present applications of the autonomous model in some industry domain. A component-based design approach was proposed by [4].

A number of works have been published on the vision of smart farm and garden. Many of them address the motivation, key elements, and features of smart farm including [5][6].

[1] This paper represents the independent work of the authors and is not sponsored or endorsed by IBM.

[2] Corresponding Author

979-8-3503-2028-2/22/$31.00 ©2022 IEEE
DOI 10.1109/CSCI58124.2022.00105

Muhtasim et al presents a realtime garden control system with security features and handles the control of watering [7].

Seo et al presents a greenhouse control system using MAPE-K model [8]. A number of works presents applications of MAPE-K in some domain, and Feng et al presents autonomous control on digital twin [9].

Our work is distinct from the related works on the scope and the approach. Our work is focused on production-level design of autonomous sprinkler system. Also, we apply two machine learning models to realize the knowledge of MAPE-K.

III. Autonomous Springker Systems

A. Conventional Sprinkler System

A conventional sprinkler system is used to control the flow of water from the meter to the sprinklers at end points. The supply of water is programmed and controlled by a 'Controller' device. The water is distributed through a network that may consist of pumps, valves, pipes, and sprinklers. Irrigation sprinklers can be used for residential, industrial, or agricultural usage. A water value then is used to open or close the water flow to sprinklers. Different types of sprinklers are used for different watering purposes as shown in Fig. 3

Fig. 3. Different Types of Sprinklers

There are limitations of conventional sprinkler systems in its effectiveness of watering.

- Problems with Underwatering and Overwatering
  The watering schedule set on a controller will automatically shut off and on water values as scheduled regardless of actual weather. Hence, it may lead to underwatering on extremely hot day and overwatering on expectedly low weather.
- Problems with not Reflecting Plant Conditions
  Different types of plants require different levels and frequencies of watering. However, the conventional current sprinkler system does not aware of plant-specific watering requirement and does not reflect the health conditions of plants. A a single uniform watering schedule is applied to all plants in a watering zone. It results in uneven growths or unhealthy conditions of plants.
- Burden of setting Watering Schedule for occasions
  To reflect the weather condition and forecast, the user has to set the watering schedule accordingly. Also, the user needs to reset the schedule for the current conditions of plants.

B. Autonomous Sprinkler System

Autonomous Sprinkler System can effectively handle the limitations of conventional Sprinkler systems. It acquires environmental condition of farms/garden using sensors and cameras, determine the optimal watering schedule, and apply the watering using IoT actuating devices. In addition, it determines the optimal level of watering for target plants and soils. It optimizes the watering schedule by considering the weather condition and forecasts.

More specifically, the system manages watering in an closed loop with MAPE-K model.

- Step 1. **M**onitoring the Environmental Conditions
- Step 2. **A**nalyzing the Contexts to Determine Situations
- Step 3. **P**lanning Watering Actions
- Step 4. **E**xecuting the Plan

The system evaluates the result of executing the plans and updates its knowledgebase accordingly. Due to the autonomous control, Autonomous Sprinkler System brings the following benefits.

- Healthier Plants and Soil
  The system optimizes the watering schedule by considering the weather and plant conditions. It eliminates the issues of underwatering and overwatering. As the result, the plants can stay healthy and higher harvest productivity is achieved.
- Reduced Consumption of Water
  By avoiding the overwatering and optimal watering schedule for the environmental condition, there will no excessive consumption of water.
- Free of Burden to set Watering Schedule
  Due to the autonomic control loop for managing the whole process of watering, users no longer set the watering schedule manually.

C. Functional Requirements

The functionality of the system is classified into the functional categories including the followings.

Hardware Device Registration: This functionality is to register hardware devices including sprinkler controller, water valves, sprinkler heads, sensors, cameras, and IoT type actuators.

Registration of Layout and Managed Objects: This functionality is to specify the layout of a target farm/ garden. The layout information is utilized by the system in locating the locations of watering zones, sprinkler heads, sensors, cameras, and actuators. The plants to be managed are also registered. With the information about type of plant, name of plant, location, and characteristics.

Monitoring of the Environment: This functionality is to acquire contexts of the environment using sensors and cameras. The acquisition of contexts enables the detailed monitoring of the plants.

Analyzing the Situation through Contexts: This functionality is to analyze the acquired contexts and determine the situation of each managed object such as plants. The analytics on the contexts can be performed in various ways, but advanced

analytic methods should be utilized to yield a high accuracy of context analysis.

Planning the Watering Schedule: This functionality is to devise a plan, i.e., watering schedule, that is the most appropriate for the determined situation. An actuation plan consists of one or more actions, and each action can be a watering action or a sprinkler calibration action.

Executing the Watering Schedule: This functionality is to execute the generated plan, i.e., performing watering according to the generated schedule. As the result, plants and soils are provided with an appropriate level of water.

Updating the Knowledge: This functionality is to manage the knowledgebase of the autonomous sprinkler system. An initial knowledge is defined with machine learning models.

IV. Design of the Framework

A. Skeleton Architecture

The target system can be well modeled with Client-Server architecture style as shown in Fig. 4.

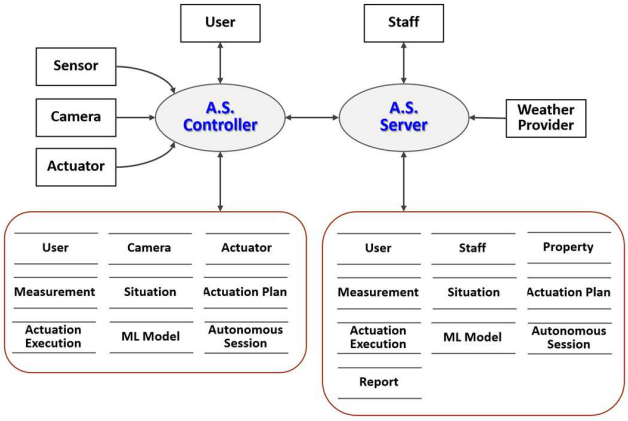


Fig. 4. Client-Server Architecture for Target System

A.S. Controller, i.e., Autonomous Sprinkler Controller, is to replace the conventional sprinkler controller. It provides autonomous control capability, and it communicates with the server for storing the acquired contexts and downloading the up-to-date knowledge used in ASC.

A.S. Server is the back-end server which is to store the data repository of the collected contexts and the applied schedules and to perform advanced analytics on the system operations. Based on the analytics results, the staffs on the server side may develop and deploy an enhanced version of *ASC* elements such as the key knowledgebase.

We applied four architecture styles in devising the architecture of the framework: Client-Server style, Layered style, MVC, and Event-based architecture styles. The resulting skeleton architecture is shown in Fig. 5.

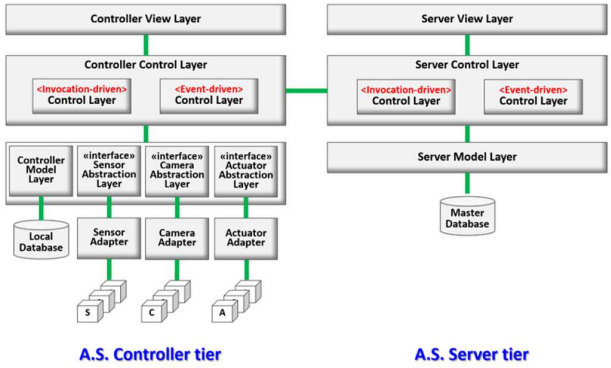


Fig. 5. Skeleton Architecture of the Framework

To make the system adaptable to various types of hardware, we define three abstraction layers: Sensor, Camera, and Actuator abstraction layers.

B. Functional Components

We model the system functionality in a use case diagram, and define several software agent actors that run in background as daemon process. This is to enable the autonomic control of the system. *Context Agent* is to acquire environmental contexts using sensors and cameras, *Situation Agent* is to determine the situations of the plants and soil, *Plan Agent* is to generate a watering plan, and *Execution Agent* is to run the watering according to the schedule.

We derived the functional components from the use case diagram. We cluster each set of relevant use cases into a functional component for high cohesion. The functional components for the controller tier are shown in Fig. 6

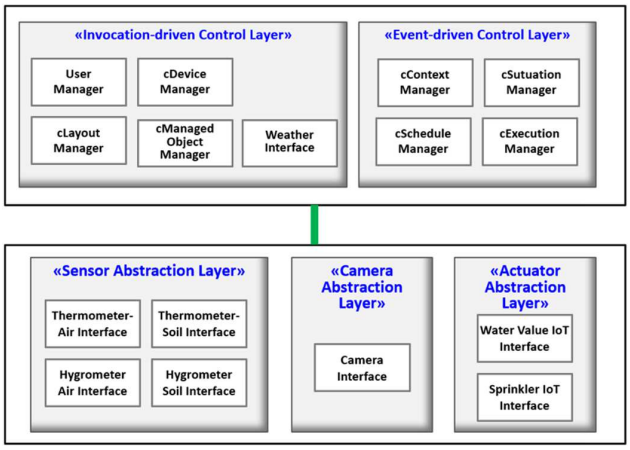


Fig. 6. Functional Components for Autonomous Sprinkler Controller (ASC)

The top two layers host the functional components, and we classify the components into 'Invocation-driven component' and 'Event-driven component. The four components in the "Event-driven Control Layer" run upon arrivals of events. A common characteristic of the 4 components is the functionality should be invoked only when an appropriate event is emitted by some other thread and arrives.

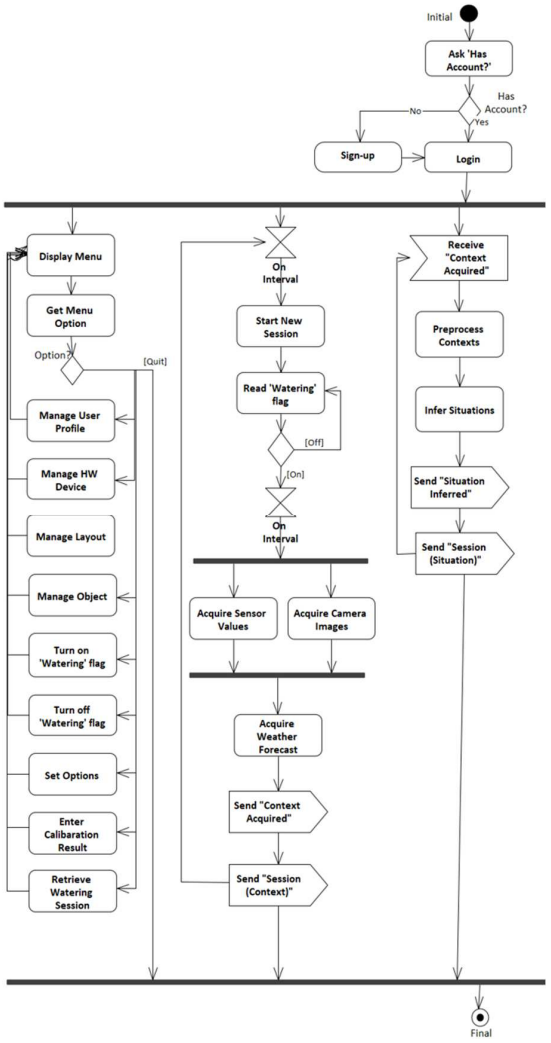
We define three abstraction layers to provide high interoperability with heterogeneous sensors, cameras, and actuators.

C. Control Flow for the Controller

The behavior view design depicts the runtime control flow of the whole system, and activity diagram can serve the purpose. The control flow for the controller (ASC) tier is shown in Fig. 7.

As shown in the figure, the control flow consists of 6 parallel threads. Only the first thread is to run the functionality with explicit invocation by users. The next 4 threads are to run each of M, A, P, and E operations of autonomous computing. The last thread is to update the knowledge from the server.

The control flow in activity diagram is shown in two parts: the leftmost 3 threads and the rightmost 3 threads as in the figure.



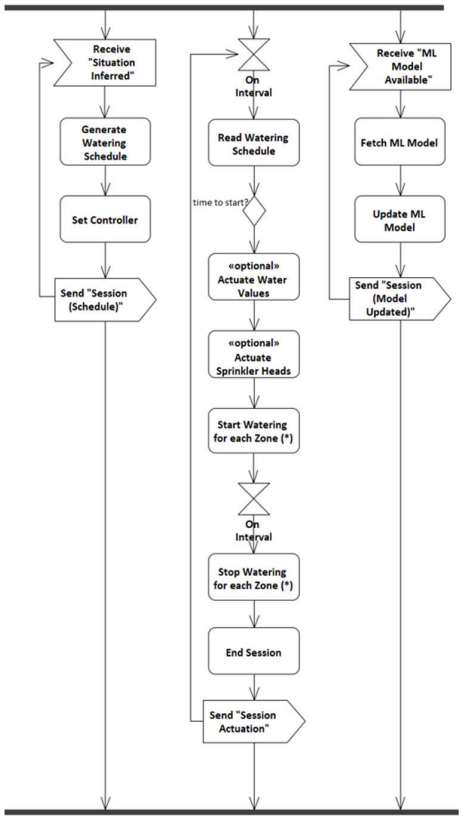


Fig. 7. Overall Control Flow of Autonomous Sprinkler Controller (ASC)

Note that all 4 threads run autonomously without users' intervention: two threads run with a timer and the other two threads run in event-driven invocation. This behavior is consistent with the MAPE-K model as in Fig. 8.

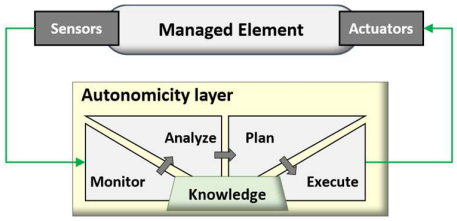


Fig. 8. Four Operations of MAPE-K Reference Model

D. Detailed Design for Inferring the Situation

The functionality of 'Inferring the Situation' is a key to enabling the autonomic control. Examples of situation can be 'Plant x: Healthy', 'Plant y: Excessively Dried', 'Soil on zone A: too wet', and 'Zone B: high heat and high moisture'.

The situation is inferred by analyzing sensor measurements, captured images of plants, and weather forecast. We apply SVM classification algorithm for the inference [10]. Hence, we first define a set of valid classes for each type of managed object.

For the situation of plants, the valid classes are (Super Healthy, Healthy, Mediocre, Unhealthy, Dead). The soil

situation is defined as a pair of (Template Level, Moisture Level) where each element can be one of the 5 classes, (Extremely High, High, Moderate, Low, Extremely Low).

For classification algorithms, we define the feature set on which the training is performed. The features for classifying plant situations are Weather Forecast, Air Temperature, Soil Temperature, Air Moisture level, Soil Moisture level, and plant condition which is inferred from captured camera images.

Using the classifiers, we devise the main algorithm for situation inference as shown in List 1.

List 1. Main Algorithm for Interring Situations

Algorithm. Inferring Situations with Classifiers

Input: *weatherForecast*, airTemperature, airMoistureLv, soilTemperature, soilMoistureLv, img

Output: *airSituation, soilSitaution, plantSituation*

```
inferSituations(..) :(...)
  // Load Classification Models
  airSituationClassifier := // To load trained Air Situation Classifier
  soilSituationClassifier := // To load trained Soil Situation Classifier
  plantSituationClassifier := // To load trained Plant Situation Classifier

  // Preprocessing Data
  weatherCat := categorizeWeatherForecast(weatherForecast);
 // Categorize input weather Forecast
  imgScaled := img/255; // Scale input image data
    // scale numerical features
    weatherCatScale := (maxWthCat - weatherCat) / (maxWthCat -
minWthCat);
    airTempScale := (maxAirTemp-airTemperautre)/(maxAirTemp-
minAirTemp);
    airMoiLvScale:= (maxAirMoiLv-airMoistureLv)/(maxAirMoiLv-
minAirMoiLv);
    soilTempScale:=(maxsoilTemp-soilTemperautre)/(maxsoilTemp-
minsoilTemp);
    soilMoiLvScale:=(maxSoilMoiLv-soilMoistureLv)/(maxSoilMoiLv-
minSoilMoiLv);

  dataAirSituation := [weatherCatScale, airTempScale, airMoiLvScale];
  dataSoilSituation := [weatherCatScale, soilTempScale, soilMoiLvScale];
  dataPlantSituation := [weatherCatScale, airTempScale, airMoiLvScale,
soilTempScale, soilMoiLvScale, img];

  // To predict the Air Situation
  airSituation := airSituationClassifier.predict(dataAirSituation);
  // To predict template level and moisture level for Soil Situation
  soilSituation := soilSituationClassifier.predict(dataSoilSituation);
  // To predict template level and moisture level for Plant Situation
  plantSituation := plantSituationClassifier.predict(dataPlantSituation);

  return airSituation, soilSituation, plantSituation;
```

E. Training SVM & CNN Multioutput Classifers for Plant Condition

Support Vector Machine (SVM) algorithm can be used to read an input observation and to generate multiple output values. In our design, it generates two types of output values.

The plant condition, i.e., the healthiness of plants can be inferred by Convolutional Neural Network (CNN) model. We trained a Fast R-CNN classifier with a configuration of convolution layers, max pooling layers, and ResNet blocks. The configurations of the model layers and relevant hyperparameters are specified in Table I.

TABLE I. CNN LAYERS AND HYPERPARAMETERS

ID	Layer	Input Size	Output Size	Kernel Size	# of Filters	Stride	Activation Function
1	Input	512*512*3	512*512*3				
2	Convolutional	512*512*3	510*510*3	(3,3)	3	1	ReLU
3	Max Pooling	510*510*3	255*255*3	(2,2)			
4	Convolutional	255*255*3	127*127*6	(3, 3)	6	2	ReLU
5	Max Pooling	127*127*6	63*63*6	(2,2)			
6	Convolutional	63*63*6	21*21*9	(3, 3)	9	3	ReLU
7	Max Pooling	21*21*9	10*10*9	(2,2)			
8	Flatten	10*10*9	900				
9	Fully Connected	900	100				ReLU
10	Fully Connected	100	25				ReLU
11	Fully Connected	25	5				Sigmoid

V. IMPLEMENTATION AND EXPERIMENTS

A. PoC Implementation

We implemented a Proof-of-Concept system in Python using libraries of TensorFlow for Fast F-CNN model, and Scikit-learn for SVM model. All the design of architecture, functional components, and their behavior design have been implemented by maintaining the high consistency with the design.

B. Experiment Site and Layout

The experiment environment was on a single-story conventional house in Eastvale, California. The house has a yard size of 8,276 square feet, and gardens are on all four sides of the house: front yard, back yard, right side yard, and left side yard. Accordingly, there are 4 watering zones as shown in Fig. 9

Fig. 9. Backyard Watering Zone of the House under Experiments

C. Hardware Configuration

We have installed a number of hardware devices for our experiments: water-proof cameras, thermometers for atmosphere and soil, Hygrometers for the humidity of atmosphere and Soil. We also acquired cameras and sensors with WIFI connectivity as shown in Fig. 10.

Fig. 10. Cameras and Sensors with WIFI connectivity

All these devices provide device drivers for remote controllability. We also installed watering actuators with WIFI-connectivity: IoT Water Values and IoT Sprinklers. The hardware configuration is shown in Table II.

TABLE II. NUMBERS OF HARDWARE DEVICES INSTALLED

Zone	Plants	Sensors	Cameras	Values	Sprinklers
Front Y.	16	8	4	2	12
Back Y.	41	13	7	3	38
Left S.Y.	7	3	2	1	8
Right S.Y	5	3	2	1	8

D. Experiment Scenarios and Results

The experiment has been conducted for 12 months period from October 2021 to September 2022. The results of a year-long experiments then were compared to those of the previous year, October 2020 to September 2021. We conducted an extensive set of experiments but presents some results of the experiments.

Throughout a year-long experiments, we acquired an average performance of was 93.5% for the SVM classifiers, and an average Intersection of Union (IoU) performance of 86% for the Fast R-CNN model.

Now, we define a metric for computing a compound plant health as the following.

$$\textit{Plant Quality} = \sum_{i=1}^{n}(\text{Criterion.i} * \text{Weight.i}) / \text{n}$$

for 'n' comparison criteria and the sum of all weights is 1.

Then, the value range of the plant quality is 0..1.

The number of underwatering and overwatering occurrences during the experiment period is shown in Fig. 11.

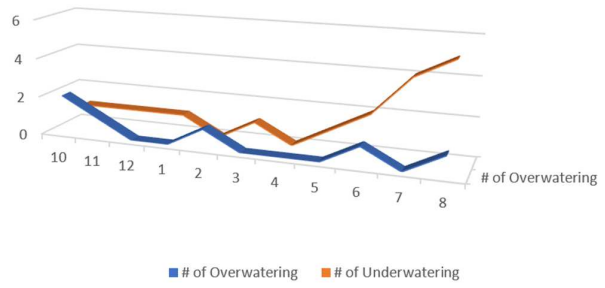


Fig. 11. Frequency of Overwatering and Underwatering

It is about 75% reduction of the occurrences compared to the previous year. The occurrence rate for the summer season is shown to be higher than others, due to the extreme heats in the region of the experiment house.

VI. CONCLUDING REMAKRS

Sprinkler is a key element for farms and gardens. However, conventional sprinkler systems have limitations: underwatering and overwatering under evolving weather, not considering plant conditions, and burden of manual scheduling.

We designed and implemented *Autonomous Sprinkler System* to remedy the limitations of conventional sprinkler systems. We collect a rich set of environmental contexts using IoT sensors, cameras, and actuators. Then, we apply MAPE-K for autonomic control.

We implemented the Knowledge with a SVM classifier for inferring the situations and a Fast R-CNN for determining plant condition. Our experiments show the system yields a considerable savings of water consumption and promotes the plant health, all without human users' interventions. The system has been effective in removing 75% of the occurrences of overwatering and underwatering in the previous year.

Our future work is to provide the fault tolerance for various malfunctions and failures of connected sensors and IoT devices.

REFERENCES

[1] Thamaraimanalan, T., et al. "Smart garden monitoring system using IoT." Asian Journal of Applied Science and Technology (AJAST) 2.2 (2018): 186-192.0

[2] S. Olawepo, A. Adebiyi, M. Adebiyi and O. Okesola, "An Overview Of Smart Garden Automation," 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS), 2020, pp. 1-6, doi: 10.1109/ICMCECS47690.2020.240892.

[3] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," in Computer, vol. 36, no. 1, pp. 41-50, Jan. 2003.

[4] S. Ouareth, S. Boulehouache and S. Mazouzi, "A Component-Based MAPE-K Control Loop Model for Self-adaptation," 2018 3rd International Conference on Pattern Analysis and Intelligent Systems (PAIS), 2018, pp. 1-7, doi: 10.1109/PAIS.2018.8598529.

[5] V. S. Kumar, I. Gogul, M. D. Raj, S. K. Pragadesh and J. S. Sebastin, "Smart autonomous gardening rover with plant recognition using neural networks", *Procedia Computer Science*, vol. 93, pp. 975-981, 2016.

[6] Shireen Nishath and B Vasundhara Devi, Smart garden management system International Journal of Engineering Research in Computer Science and Engineering (IJERCSE), vol. 5, no. 3, pp. 6-10, 2018.

[7] Muhtasim, S. R. Fariha and A. M. Ornab, "Smart Garden Automated and Real Time Plant Watering and Lighting System with Security Features", *2018 International Conference on Computing Power and Communication Technologies (GUCON) Galgotias University*, pp. 676-679, 2018.

[8] Y. Seo, et al, "Design of a smart greenhouse system based on MAPE-K and ISO/IEC-11179", 2018 IEEE International Conference on Consumer Electronics (ICCE), 2018.

[9] Hao Feng;Cláudio Gomes;Santiago Gil;Peter H. Mikkelsen;Daniella Tola;Peter Gorm Larsen;Michael Sandberg, "Integration Of The Mape-K Loop In Digital Twins", 2022 Annual Modeling and Simulation Conference (ANNSIM), 2022.

[10] "Support Vector Machines", scikit-learn documentation". Archived from the original on 2017-11-08.