都市型農業のサービス産業化を実現する水耕栽培システム

佐藤証[1]

概要：都市部の空きスペースを活用し，農業のサービス産業化を実現するため，小型水耕栽培プランタの開発と栽培実験を進めている．農業初心者でも気軽に本格的な栽培が楽しめるよう，水耕栽培用センサの低コスト化と遠隔管理システムの実装を行った．センサデータをモニタし，ポンプを制御するのに，IoT 用の軽量プロトコル MQTT を用い，TLS/SSL で通信の暗号化も行っている．数値データのモニタの他に，USB カメラを用いた動体検知による画像撮影機能も実装した．さらに，スマートフォンや PC だけでなく AI スピーカーのサポートなど，インタフェースの充実も図っている．

1. はじめに

筆者は土地を耕し作物を生産するこれまでの農業に対して，都市の屋上やマンションのベランダなどの空きスペースを利用した人々が楽しむ農業を提案し，その実現に向けて水耕栽培装置の開発と栽培実験を行っている[1]．そして，オフィスビルや病院，小学校などに導入し，屋上緑化や憩いのスペース創出，食育への活用，地産地消から自産自消へ，そして地域連携による 6 次産業化も進めている．

このような楽しみを目的とした都市型農業では，生産性の追求は重要ではない．しかし，農業の知識や経験のない初心者をメインユーザとしているため，栽培支援が必要である．そのために都市部のビルやマンションに点在する中小規模の施設の状況を，サービス提供者が毎日直接チェックすることは現実的ではない．IoT 技術による遠隔管理システムは有効な解決策であるが，大規模植物工場などで用いられている農業用センサは非常に高価で，個人や小規模施設での利用は難しい．農業用センサは最適な環境制御のため，高い精度と信頼性が要求される専用装置であることが高価になる大きな要因である．

そこで本稿では，IoT デバイスとして高性能化と低価格化が急速に進むマイコンボード Arduino や Raspberry Pi，そしてオープンソースのソフトウェアなどを活用し，個人でも気軽に楽しめる水耕栽培システムを構築する．また，Arduino の子基板として水耕栽培用のセンサモジュールも開発した．目的とする楽しむ農業ではセンサの精度と信頼性にある程度の妥協が許されるため，シンプルな回路構成にすることでコスト削減も実現している．この水耕栽培システムのハードウェアと制御ソフトウェアについて以下で紹介する．

2. 小型水耕栽培装置

どこにでも簡単に設置可能で土を使わずに本格的な果菜類の栽培が楽しめる，(有)上野園芸の空宙栽培方式 [2]をベースに製作した小型水耕栽培装置の外観と，その基本構造を図 1 に示す．立てたパイプに植物の株を挿し，垂れ下がった根に，タンク内の液肥をポンプで汲み上げてシャワーのよう降らせている．邪魔になるものがない根は自由に伸び，酸欠にならず，水耕栽培特有の夏場の根腐れもない．図 2 は大学屋上でのトマト栽培の様子で，フルーツトマトは糖度を上げるのに水の制限が重要であるが，ポンプの駆動時間の調整でそれを簡単に行うことができる．

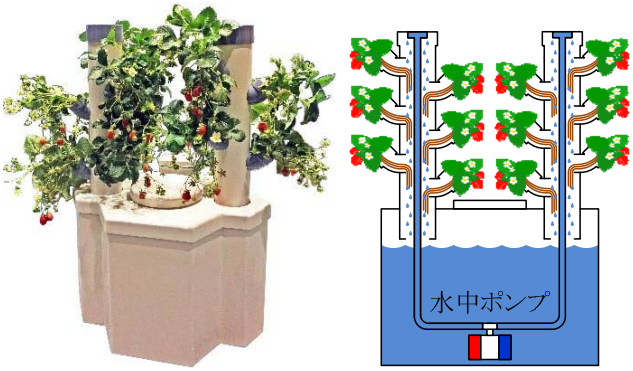

図 1　イチゴを栽培中の小型水耕栽培プランタとその基本構造

図 2　屋上でのトマトの栽培

3. センサモジュール

図 3 に水耕栽培用センサモジュールを示す．端子に各種計測用デバイスを接続して，液肥濃度（EC 値），水位，水温，照度，温湿度などを取得し，また二つのリレースイッチで循環ポンプの駆動や追肥の制御を行う．図 4 はセンサモジュールを搭載した小型プランタで，タンクの中に伸びている細長い緑色のスティックは，液肥濃度と水位を計測する基板電極である．タンク内の中央にはホースを接続した循環ポンプを，左側には自動注水用のボールタップを設置している．

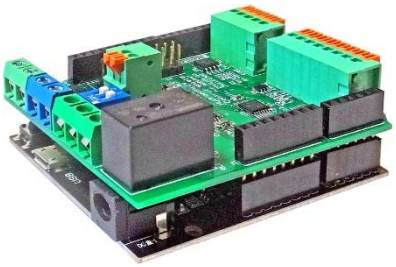

図 3　水耕栽培用センサモジュール

図 4　センサモジュールを実装した小型プランタ

図 5 は液肥濃度計測用の発振回路で，濃度によって変化する電極間の液肥の電気抵抗を，発振周波数として取得する [3]．周波数計測には ATmega328 の周波数カウンタを用いる．電極の溶出を避けるため，計測時だけ回路に交流を流しているが，発振停止時にも同じ液肥に浸かっている水位計測用の電極との間に電位差が生じると直流電流が流れる恐れがある．これを避けるために，計測時以外は図中の“制御”信号で 3 ステートバッファの出力をハイインピーダンス状態にしている．

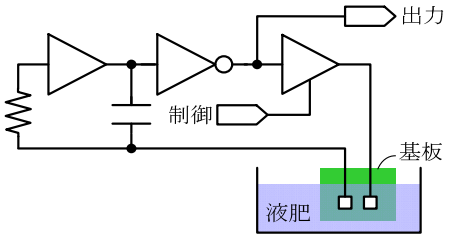

図 5　液肥濃度計測回路

図 6 は 3.3V の電源電圧を，シリアルに接続した 9 つの抵抗で分圧して水位を計測する回路である．水位の上下により，GND レベルに接続される抵抗の数が増減して，出力電圧が上下し，それを ATmega328 の 10 ビット A/D コンバータで計測する．出力電圧は 8 個の電極が全て液肥に浸かって GND につながっている状態の 0V から，全ての電極がオープン時の 3.3V まで，9 つのレベルを検出することができる．なお，この回路も電源と GND に 3 ステートバッファを挿入して，計測時以外に液肥内で直流が流れるのを防いでいる．

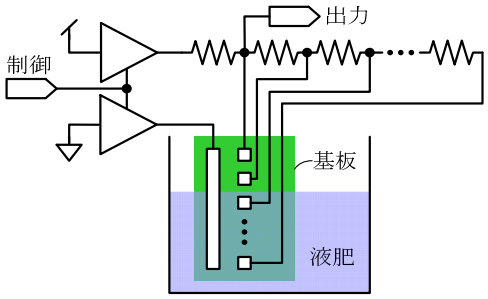

図 6　水位計測回路

水温は温度によって電気抵抗が変化するサーミスタを，照度は明るさによって電気抵抗が変化するフォトダイオードを用い，各素子にかかる電圧を A/D コンバータで計測している．また，温度と湿度の計測には，市販の安価な温湿度センサ [4]を用いた．リレーはメカニカルスイッチとパワーMOSFET の二系統を有し，いずれも駆動しているポンプなどの状態をモニタする電流センサを付加している．親基板には Wi-Fi マイコンモジュール ESP-WROOM-32 [5]を実装した Arduino 互換ボードを利用し，センサデータを無線で送信している．

センサモジュールは上記のようにシンプルな構成としてコストを低減しているが，計測したデータは周波数や電圧なので，これを EC 値，水位，水温などに変換する必要がある．この変換は Arduino 側では行わず，次節で説明するサーバの Raspberry Pi が実行する．

4.　遠隔管理システム

センサデータは Wi-Fi ルータ経由で，サーバの Raspberry Pi に送信され，外出先からはスマートフォンなどでモニタしている．この場合，グローバル IP を付与した Raspberry Pi に Web サーバを立て，http プロトコルでデータの送受信を行うのがシンプルである．しかし，栽培環境のモニタに加えてポンプの遠隔制御などを行う場合は，Raspberry Pi と同じ LAN 内にクライアントのセンサモジュールが置かれていないと，センサモジュールにもグローバル IP を割り当てる必要がある．そこで本システムでは，双方向通信時にクライアントへのグローバル IP 付与が不要で，http プロトコルよりも軽量な IoT 向のプロトコル MQTT (Message Queuing Telemetry Transport)を用いた．これにより，月額 500 円程度のデータ専用格安 SIM など，グローバル IP を持たないサービスを用いることができる．

MQTT は一旦接続した TCP セッションを切断せず，毎回のハンドシェイクが不要なため，http よりもリアルタイム制御に適している．また，通信の品質 QoS を 0~2 の 3 段階で設定できるため，大きく変化しないセンサデータは低品質の QoS0 を，ポンプ制御など重要なデータは QoS2 を用いて通信するといった設定も可能である．

しかしながら，軽量化のために最低限のプロトコルだけ

を実装した MQTT は，暗号化などのセキュリティ機能を有していない．個人が利用しているプランタ周辺の温湿度や照度などのデータはある意味プライベートな情報と言える．また，ポンプなどの制御を第三者に勝手に操作されてしまう危険も含んでいる．そこで今回の実装では，MQTT のパケットを TLS/SSL で暗号化した．このネットワークの構成例を図 7 に示す．

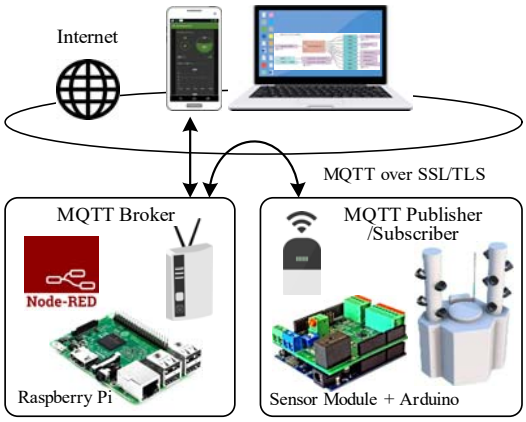

図 7　水耕栽培システムのネットワーク構成例

センサモジュールを実装した Arduino には MQTT クライアントが実装されており，Publisher として計測した各種データを 3G ルータ経由で Broker の Raspberry Pi に送信する．ユーザはスマートフォンや PC などで Broker にアクセスしてセンサデータをモニタし，また Subscriber でもあるセンサモジュールに制御データを送信する．それらモニタや制御のプラットフォームは，Raspberry Pi 上に実装した Node-RED [64]で行っている．Node-RED は Node.js で開発されたフローベースのプログラミング環境で，図 8 のように Node モジュールをつないで，IoT デバイスの制御や Web サービスなどを実現する．

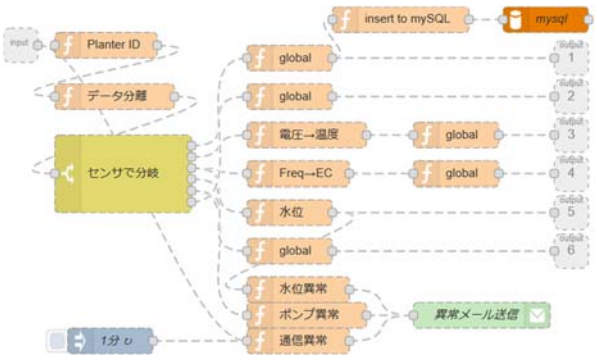

図 8　水耕栽培装置のセンサデータの処理フロー

また，GUI は Node-RED ライブラリの Dashboard [7]を用い，図 9 のような画面で各種センサのモニタやポンプの制御をリアルタイムで行っている．Dashboard 上のグラフのデータは，指定表示時間を過ぎると順次消えていくので，後から解析ができるようにオープンソースのデータベース MySQL [8]でサーバに保存している．またタンク内の水がなくなったり，ポンプが停止したり，通信が長時間途切れるなどの異常を検知した場合には，事前に指定したアドレスにメールを送信する．

図 9　Dashboard によるモニタ・制御画面

遠隔管理による栽培では，センサデータに加えて植物の状況を画像で確認することも重要である．本システムはどこにでも気軽に設置できるよう，安価な低通信速度のネットワーク環境でも動作することを前提にしているため，高解像度の動画を送信することは困難である．しかし，通常は植物が短時間で急激に成長することはなく，数～数十分に一回静止画を送るだけでも十分である．その一方で，撮影のインターバルで何か外的な要因による大きな変化が起きたことが検知できないのも問題である．そこで静止画を，指定した時間間隔で撮影して送信するとともに，フレーム間差分による動体検知を行って大きな変化が生じたときにはその画像を送信する機能も実装した（図 10）．

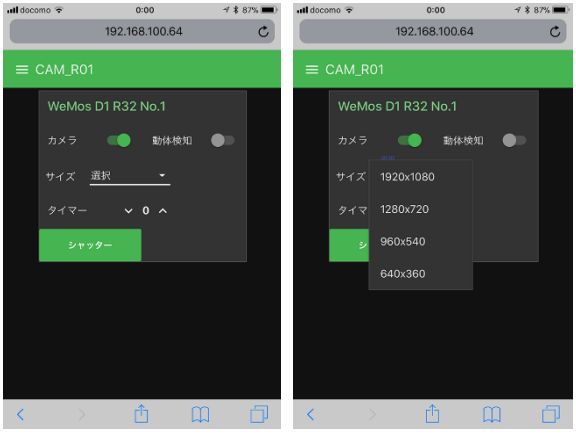

図 10　カメラの制御画面

カメラによる撮影と動体検知および jpeg 画像への圧縮処理などは，Arduino では負荷が大きく実装できないため，USB カメラや専用カメラを接続した Raspberry Pi で行っている．サーバ用には主に，64 ビットプロセッサを実装した Raspberry Pi 3 を用いているが，撮影用には安価な Raspberry Pi Zero などを用いることも可能である．現在，撮影した画像は右下にタイムスタンプを入れて，どこからでもアクセスできるようにクラウドサーバにアップロードしている．

しかしながら，撮影間隔が短いと多数の画像がアップロードされ，それを一つ一つ確認することは困難である．そ

こで，静止画をコマ送り動画のようにつないだタイムラプス動画に圧縮し Web ブラウザ上で再生する方法も実装中である．動画の生成はサーバで行うか，あるいはカメラ側の Raspberry Pi で製作し，撮影を行わない夜間の通信帯域を利用して転送することもできる．なお後者の場合，日中に動画を確認するためには Raspberry Pi に Web サーバを立てる必要があり，外部からアクセスするためにグローバル IP を付与する必要がある．そこで本システムでは，動的割当ての IP アドレスとそのホスト名を，逐次対応させるフリーの DDNS (Dynamic Domain Name System)サービスの活用も検討している．

上記のような様々な機能を有する水耕栽培システムのインタフェースとして，スマートフォンや PC だけでなく，AI スピーカーの導入も進めている．ハードウェアはサーバの Raspberry Pi に AIY Voice Kit [9]のマイク，スピーカー，LED ボタンを付加し，音声認識システム Julius [10]および音声合成システム Open JTalk [11]によって日本語化を行っている．現在，センサデータの読み上げや異常状態の通知などの基本機能を実装しているが，水耕栽培に限らず生活の中に溶け込むような機能の充実も図っていく予定である．

5. おわりに

本稿では，都市部で楽しむ農業を実現する水耕栽培システムの研究開発について述べた．個人でも気軽に楽しめることを目的に，低コストの水耕栽培用センサを開発し，汎用マイコンボードやオープンソース，そして低容量通信サービスの利用による遠隔管理システムを構築した．また開発と並行して，事業化に向けた様々なサービスビジネスの検討も進めており，今後，それらについても報告して行きたい．

参考文献

[1] 佐藤証: 新たな都市型農業のサービス産業化, マルチメディア, 分散, 協調とモーバイル (DICOMO2017) シンポジウム, 6C-1, pp.1211-1214 (2017 年 6 月).

[2] 有限会社上野園芸: 空宙トマト
http://www.uenoengei.com/

[3] T. Nishimura, Y. Okuyama, A. Matsushita, H. Ikeda, and A. Satoh: A Compact Hardware Design of a Sensor Module for Hydroponics, IEEE 5th Global Conference on Consumer Electronics (GCCE2017), OS-ICE(1)-4 (October 2017).

[4] Adafruit: DHT11, DHT22 and AM2302 Sensors.
https://learn.adafruit.com/dht/overview

[5] Espressif Systems: ESP-WROOM-32
https://www.espressif.com/en/producttype/esp-wroom-32

[6] Foundation: Node-RED - Flow-based programming for the Internet of Things.
https://nodered.org/

[7] node-red-dashboard.
https://flows.nodered.org/node/node-red-dashboard

[8] MySQL.
https://www.mysql.com/

[9] AIY Voice Kit.
https://aiyprojects.withgoogle.com/voice

[10] Julius.
http://julius.osdn.jp/

[11] Open JTALK.
http://open-jtalk.sourceforge.net