Raspberry Pi

növénygondozó rendszer

Készítette: Sári Bence

Bevezetés

Évről évre körülbelül 75 millió fővel növekedik a föld népessége, amely becslések szerint 2025re eléri a 8 milliárdot, de a lakosság növekedésével párhuzamosan csökken az egy főre jutó erőforrások nagysága.

A lakosság növekedését a mezőgazdaságnak is követni kell, annak ellenére, hogy a környezetszennyezés hatására történő globális felmelegedés, a sok helyen egyre kevésbé elérhető édesvízforrások és az időjárás egyre szélsőségesebb mivolta csak nehezíti a termelést. Egyre melegebbek a nyarak, enyhébbek a telek, tavasszal is többször fagypont alá csökken a hőmérséklet, és ezek a hatások évről évre egyre erősödnek.

Míg ha egy gazdaság elbukik évente pár millió forintot egy egy estés fagy miatt, az nem okoz világméretű problémát, de ha nem védekezünk az ilyen negatív hatások ellen a közeljövőben, az élelmiszerhiányt okozhat könnyedén.

Ahhoz, hogy ki tudja elégíteni a növekvő igényeket a mezőgazdaság, egyre hatékonyabban kell termelnie.

Az ilyen hatékony termelést két módszerrel lehet biztosítani:

1. növények génmódosítása
2. precíz, kimért termelés, akár robotokkal.

Génmanipuláció

A génmanipuláció egy olyan eljárás, melynek során a DNS-szerkezet egy részét kivágják és ezt más génekkel helyettesítik.

Ez már évezredekkel ezelőtt elkezdődött, amikor a haszontermő növényeket kiválogattuk és csak azokat termesztettük.

Növénynemesítéssel erősíthető egy növényfaj, de génmódosítással látványosabb eredmény elérhető.

Eredménye: Nagyobb ellenálló képesség a vízhiány, kártevők, kevés napsütés, betegségek, vegyszerekkel szemben.

Veszélyei sokak szerint vannak, de tudományosan nem támasztották alá őket még.

Robotizálás

Pontos mennyiséget szolgáltat a növénynek vízből, tápanyagból vagy fényből.

Csak akkora területet öntözünk, amekkora szükséges.

Tápanyagot lehet úgy adagolni, hogy a növény egyenletesen tudjon fejlődni.

Ha kevés napfény éri a terményeket, fluoreszcens vagy LEDes fényekkel ki lehet egészíteni a napsütötte órák számát.

Pontosan tudjuk rögzíteni, hogy a növények milyen feltételeknek megfelelően hogyan fejlődött, így később egy tökéletes tervezetet lehet előállítani a termeléshez.

A génmanipuláció most az a terület, amiben bíznak, hogy biztosítani tudja azokat a növényeket majd később, melyek sokkal ellenállóbbak lesznek a szárassággal, baktériumokkal, hírtelen nagy mennyiségű csapadékkal szemben.

Már most is vannak olyan kísérletek, amelyekben olyan szárasságtűrő növényekkel kísérleteznek, melyek, ha nem kap folyadékot, hibernálja magát, majd ha újra folyadék éri, fejlődik tovább, ott ahol abba hagyta, és a céljuk a kísérleteknek, hogy ezeknek a növényeknek, azon génjeit, amelyeknek köszönhető az ilyen szárasságtűrő tulajdonságuk, áthelyezzék különböző haszontermő növényekbe, így olyan földrajzi területeken is lehetne termelni, ahol most lehetetlennek tűnik.

Viszont sok régióban nem szeretik a génmódosított terméseket, így van ahol csak fel kel tűntetni, hogy génmódosított terméket rejt a csomagolás, viszont van olyan is, ahol a kormány nehezíti a génmanipulált növények forgalmazását, termesztését.

Ezáltal ha nem szeretnénk génmódosított növényt termelni, de ilyen ütemben folytatódik a globális felmelegedés, kénytelenek leszünk sokkal intelligensebb mezőgazdasági gépeket használni, mivel ha kevés csapadék esik, most is van lehetőség az öntözésre, viszont a jelenlegi eszközökkel kevés víz jut ténylegesen a növényre, és rengeteget szórunk a körülötte lévő földre, ahol így haszontalan gaz nő ki, amit később permetszerrel írtunk ki általában. Ez a korlátozott mennyiségben elérhető édesvízkészletek miatt nem folytatható sokáig.

A mezőgazdasági eszközök időről időre egyre modernizálódnak, így a robotika is egyre nagyobb részt fog magáévá tudni a termelésben, ahogy azt az iparban is teszi. (pl. KUKA)

Rengeteg próbálkozás, és késztermék is készült már abból a célból,
hogy olyan eszközt hozzanak létre, amely a beavatkozás nélküli, vagy csak kevés beavatkozást igénylő termelést tud biztosítani.

Mivel a termelők többsége hagyományos módon próbál termelni most, ezeknek az ára még nagyon magas, de később, ha a termés minősége annyira rossz lesz, hogy küszködni fognak a jelenlegi módszerekkel, akkor egyre többen fognak olyan fejlesztésbe fektetni – mind otthon illetve gazdaságokban - amely robozitált gazdálkodást valósítja meg akár fóliasátorban, üvegházban, vagy akár csak egy szimpla termőföldön.

Jelenleg is működnek olyan üvegházak, ahol télen-nyáron folyik a növénytermelés, de ezekből kevés van még, mivel ez csak nagy befektetés árán tud létrejönni. Egy átlag mezőgazdásznak nincs erre tőkéje sajnos.

Szerintem ahhoz, hogy a mezőgazdászok számára is elérhetővé tegyük a termelést okosító eszközöket, olyan pályázatokat, vagy programokat kellene létrehozni, melyekben mezőgazdászok mérnökökkel társulhatnának, a közös sikerért.

Ezért nekem már régebb óta eszembe volt, egy olcsó növénygondozó rendszer megépítése, viszont ahhoz, hogy könnyen módosítható, felügyelhető lehet, a jelenlegi eszközeim nem voltak elegek.

Viszont a Raspberry Pi adott egy olyan alapot az egésznek, amit könnyen elérhetek bármikor, bekapcsolva hagyhatom napokig, tudja a pontos időt, van rajta internetkapcsolat, nagy memóriába tudja menteni az adatokat, így gondoltam megpróbálom megvalósítani ezt az ötletem a Raspberry és Arduinoval együttesen.

A projektem végül pedig így valósult meg:

Arduino:

Úgy döntöttem miután utána olvastam a Raspberry GPIO használatának, hogy inkább használok egy arduino-t az érzékelők, és beavatkozók összekötésére, mert nem szerettem volna kicsit is veszélyeztetni az Raspberry-t, a korántsem tökéletes elektronikai tudásom miatt. Így ha bármit rosszul kötök be, egy olcsóbb Arduino veszik csak oda, nem pedig egy drágább Raspberry, ami amúgy sem az enyém.

Így projektemben első lépésként az Arduinoval kapcsolatos részt csináltam meg, ami első sorban az érzékelők beszerzéséből állt.

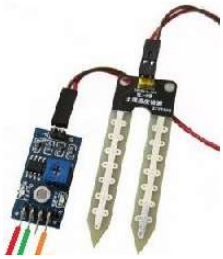
Érzékelők:

Összesen 5 darab érzékelőt szerettem volna használni, és egy beavatkozót.

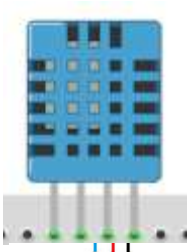
Photoresistor: Ez egy olcsó ellenállás, amely fény hatására csökkenti az ellenállás nagyságát, ez volt az első érzékelő, amit kipróbáltam a tesztek során.

Eső érzékelő: Arra szolgált volna, hogy ha esni kezd az eső, akkor a rendszer ne öntözzön, hanem csak várjon addig amig el nem áll az eső, viszont nem sikerült olyan vízálló házat építeni, amit kint mertem volna hagyni tényleges esőben, így nem sikerült élesben kipróbálni, viszont vízcseppekkel működött tökéletesen.

Talajnedvesség mérő: A projekt legfontosabb eleme, ez mutatja meg, hogy mennyire nedves a talaj, és így tudtam szabályozni, hogy mennyi nedvességet kapjon a növény. Ebből sajnos csak 1 darabom van jelenleg, viszont később, ha komolyabban szeretnék növényeket ilyen rendszerrel gondozni, elengedhetetlen belőle több is, vagy robotkarral a mozgatása.



Az érzékelőkből sajnos nem sikerült mind felhasználnom:



Dht11: mivel a raspberry-n az arduino nem bírta az új könyvtárat kezelni, ami kellett volna a DHT11 érzékelőhöz. Ez a páratartalom és hőmérsékletet mérte volna. Az arduino kódban még benne is van a rész ami windows alatt működik tökéletesen, viszont a dht11-el kapcsolatos részeket ubuntu alatt ki kellett kommenteznem.

Vízszint: a másik, amit nem használtam fel, a vízszintet mutatta volna, ami működött is tökéletesen, csak mivel pet palackból oldottam meg az öntözést, nem tudtam belerakni úgy a vízszintmérő-t, hogy ne lepje el a víz teljesen, úgy pedig fals adatot közölne. Ha más víztározót használnék, jobban fel tudnám használni.

Beavatkozó:

A terv egy olcsóbb szivattyú beszerzése volt, amit végül egy autóalkatrész kereskedésben sikerült megtalálni, így az öntözésért felelős egység egy régi fajta Skoda ablakmosófolyadék pumpája volt a projektben. Eleinte 5 volton teszteltem, amin habár tudta a vizet pumpálni, lassú volt, később 12 volton működött, akkor pedig már elég nagy sebességgel tudta a vizet pumpálni.

Viszont ezért kellett az Arduinonak plusz tápfeszültség, mivel a motor teljesen elszívta a meglévőt, így végül egy 12V DC - 1A adapterrel oldottam meg az áramellátást.

Programkód:

Ezekhez írtam egyszerű tesztelő programokat, amik soros portra küldik az érzékelőkről beolvasott analog eredményt, így biztosra mehettem, hogy mindegyik működőképes.

Majd ezeket a programokat összefűztem, és így született meg a későbbi kód alapja.

Raspberry beüzemelése:

Az RPI-t miután elindítottam, és hálózatba kötöttem, az Nmap nevű programmal kerestem meg, hogy milyen IP-címet kapott a routertől, miután ez megvolt, a Putty programmal ssh-n keresztül letöltöttem, és telepítettem a vnc szervert az RPI-re:

```
sudo apt-get install tightvncserver
```

Ezután elindítottam a szervert:

```
tightvncserver
```

Majd beállítottam az aktuális felbontást, ami vagy 1080p volt vagy 720p

```
vncserver :1 -geometry 1920x1080 -depth 24
```

Aztán pedig vagy asztali gépről, vagy laptopról csatlakoztam rá, és használtam a rendszert.

Raspberry - Arduino:

Mivel még ez előtt se Linux alapú rendszerekkel, se Raspberry-vel nem volt dolgom így eleinte meggyűlt vele a bajom. Először is nem tudtam, hogy lehetne a két eszköz közti kommunikációt felállítani. Az meg volt, hogy USB-n keresztül szerettem volna, hogy üzenjenek egymásnak, de azon kívül, hogy soros porton kellene ezt megoldani, nem tudtam semmit.

Az első lépés az RPI-re való Arduino szoftver telepítése volt:

```
$ sudo apt-get update
$ sudo apt-get install arduino
```

Első frissíti a rendszert, második pedig telepíti az arduinohoz szükséges programokat.

Ezután sikerült már az arduinot a raspberry-ről programozni, ekkor viszont még nem tudtak parancsokat adni egymásnak, ezért utána olvastam, és a soros portot ajánlották, hogy raspberryn pythonban megírt kódban is használhatom, illetve arduinom már jelenleg is soros portra küldte az adatokat az érzékelőkről.
így hát telepítettem a pyserial-t:

```
$ sudo apt-get install python-serial
```

Ez után pedig jött a python kód írása.

Python 2:

Ez előtt soha nem írtam még python kódot, így mindenek előtt el kezdtem az alapokat tanulgatni, hogy hogy néz ki egy egyszerű kód. Viszont sajnos sokszor előfordult, hogy egyszerű tagolásokon akadt fent a kód, amit én nem láttam át még, így nem is tudtam mi a hiba, de pár óra gyakorlás után már többnyire működő kódokat tudtam írni.

Első körben már a soros kommunikációval próbálkoztam a két eszköz között, mivel az arduino már küldte az adatokat, gondoltam írok egy egyszerű adat váró/fogadó kódot.

Legelőször a soros portot kellett beállítani

```
import serial
ser = serial.Serial('/dev/ttyACM0', 9600)
```

A ***ttyACM0*** jelenti az eszköz nevét, aminek megtalálására sokan konzolparancsot ajánlottak, viszont később rájöttem, hogy az arduino környezet is írja, az aktuális eszköz nevét, így ha ez megváltozott,(véletlen leállások miatt pedig sokszor megváltozott) akkor onnan könnyen átírhattam.

A ***9600*** pedig az átviteli sebesség, aminek Arduino-n is ennyinek kellett lennie.

A program későbbi részeiben pedig erre a soros beállításokra a 'ser.' paranccsal tudtam hivatkozni, pontosan a ser.readline() ez beolvas, ser.write() ez ír adatot.

Mivel a cél az volt, hogy a kapott adatot lementse a program, a későbbi felhasználásra, így miután az első adatátvitel sikerült, el kezdtem a fájlba írás részt megírni.

```
import io
filename = „fájlnév"
target = open(filename, 'a')
target.write(írnivaló)
target.close()
```

Importáltam az input output könyvtárat, megadom, hogy hova írjon a program, és azt, hogy hozzáfűzzön, vagy felülírja az előző adatokhoz a program, mert elég sok ideig a 'w' parancsot használtam, ami felülírta az előző adatokat, de végül megtaláltam, hogy az 'a' a meglévő adatokhoz hozzáfűzi az újakat.

Az írás után pedig lezárom a fájlkezelést.

A program alapja így már szinte kész volt, már csak az aktuális időt szerettem volna az adatok mellé írni, illetve az egészet egy programba beírni.

```
import datetime
import time
now = datetime.datetime.now()
idő = „$s:$s"$(now.hour,now.minute)
```

Ezekkel a sorokkal már pontosan meg tudtam adni az aktuális időt, nap, óra, perc pontosan.

Kész programok:

Az arduinora írt kódot, és a pythonban írtat össze kellett párosítani, mivel azt szerettem volna elérni, hogy a rendszer csak nappal működjön. Ennek biológiai oka van, ugyanis este a növények is alszanak, így a legjobb számukra ha éjszaka nem kapnak folyadékot, és napfényt se. Ezért úgy írtam át az arduino kódot, hogy várjon a raspberry oldalról egy jelet, ami vagy 0 vagy 1, és 1-es esetén futhat le a kód csak, ami beolvassa az adatokat, illetve öntözi a növényt.

Ezt egy függvénnyel oldottam meg, ami 0-t ad vissza ha este van, és 1-t ha nappal

```
def mehet(ora):
    x=0
    if now.hour>7 and now.hour <20:
        x=1
    return x
```

Így a program futás közben folyamatosan ellenőrzi, hogy futhat-e, és ha futás közben este lesz, akkor rögtön megszakítja az adatok olvasását és fájlba írását, majd vár egészen nappalig.

Arduino pedig nem kezdi el küldeni az adatokat egészen amíg nem kap 1-est a soros porton.

A fájlba írás a következőképpen néz ki: megadott fájlnév után fűzi az adott napot, így minden nap új fájlt kezd, ezután az adatok íráskor így kerülnek bele:

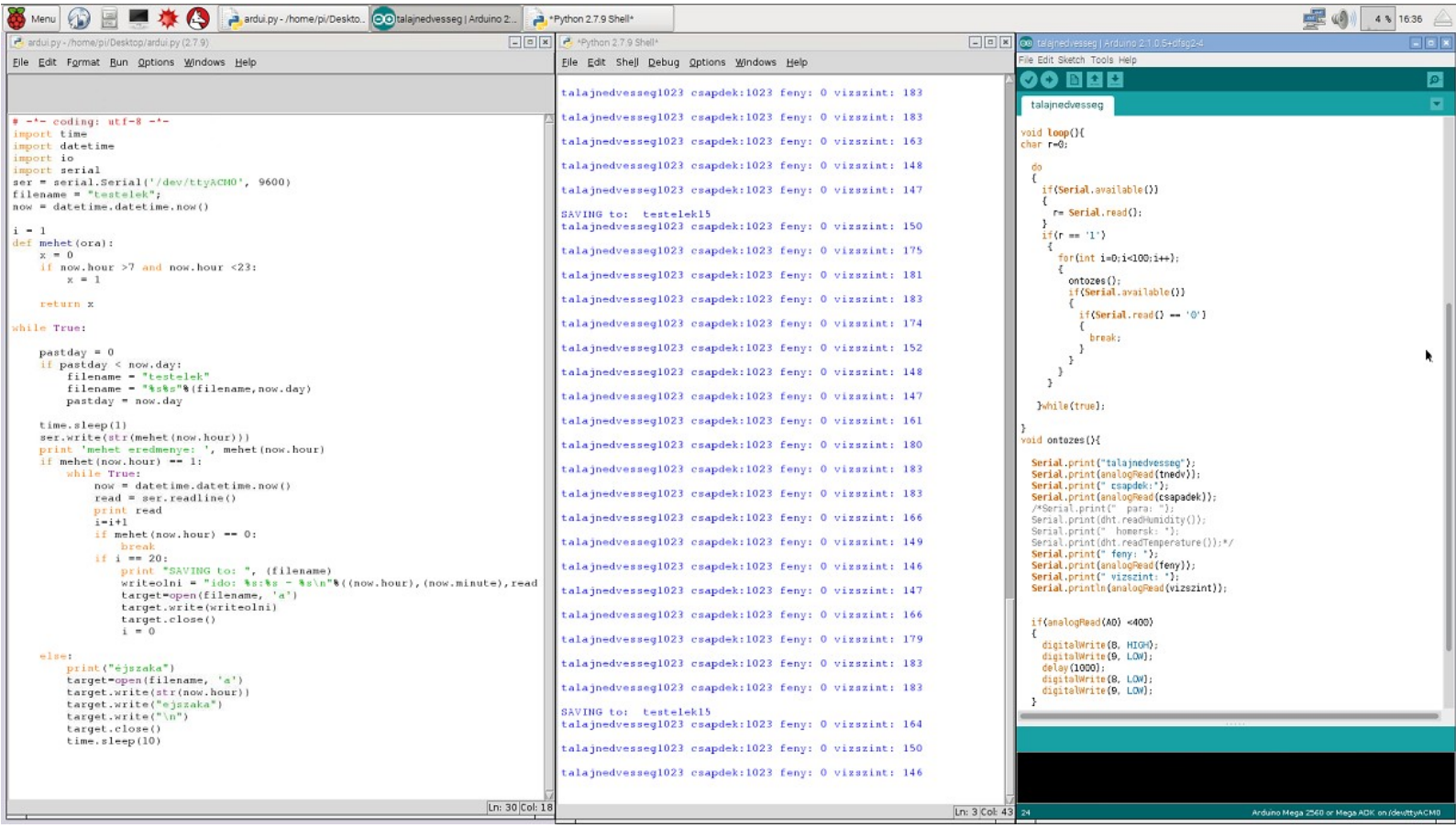
ido: 13:51 – talajnedvesseg: 358 csapdek: 1022 feny: 614 vizszint: 525

itt jelenleg délután volt, növények megöntözve, eső nélkül, árnyékban.

mivel a pythonban a sorok behúzása fontos szerepet játszik úgy pedig nem tudtam kimásolni a programot hogy ez a tagolás meg is maradjon, nem másolom ide be a programot, csak mellékletbe csatolom azt.

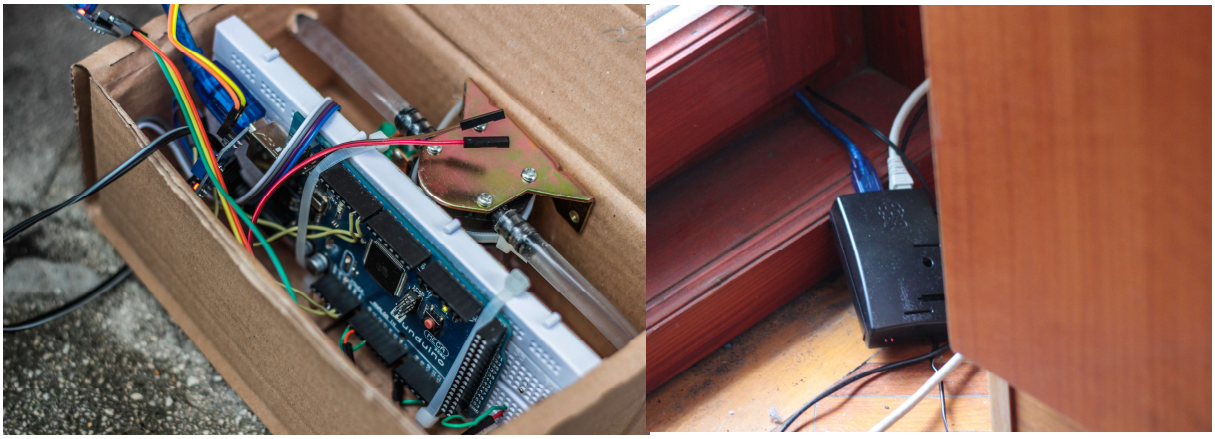
Így nézett ki működés közben a rendszer:

Bal oldalon Python kód középen az olvasott adatok jobb oldalon az Arduino



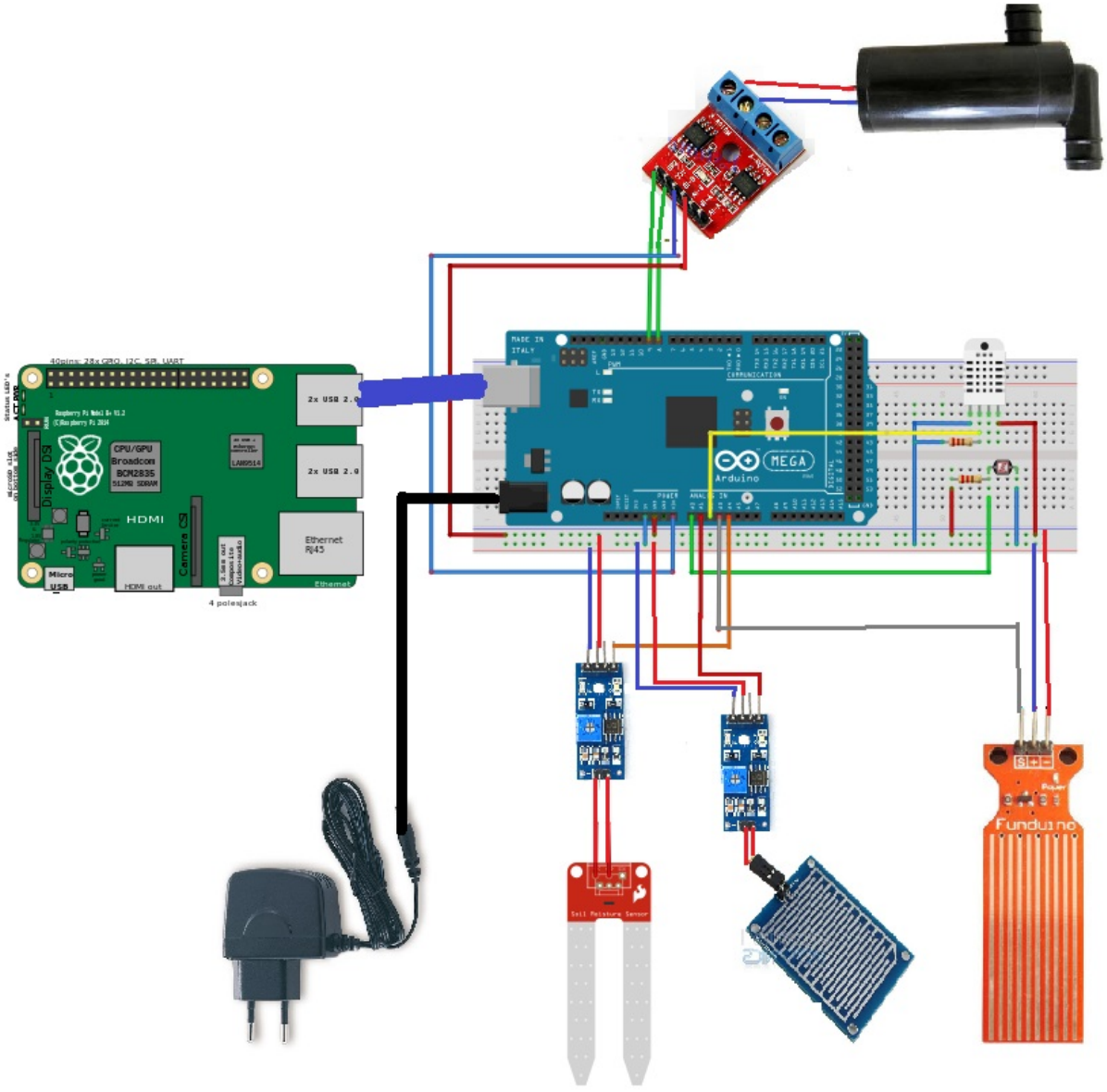
Összeszerelés:

A rendszer összeszerelve úgy nézett ki, hogy egy dobozba került a Protoboard az Arduino Mega2560, a szivattyú, a photoresistor, az eső érzékelő, a dobozon kívülre került a talajnedvesség, és vízszint érzékelő. Ezek mind a lakáson kívülre, az erkélyünkre kerültek, a raspberry, és az adapterek pedig a házon belülre, hogy semmi bajuk ne eshessen.

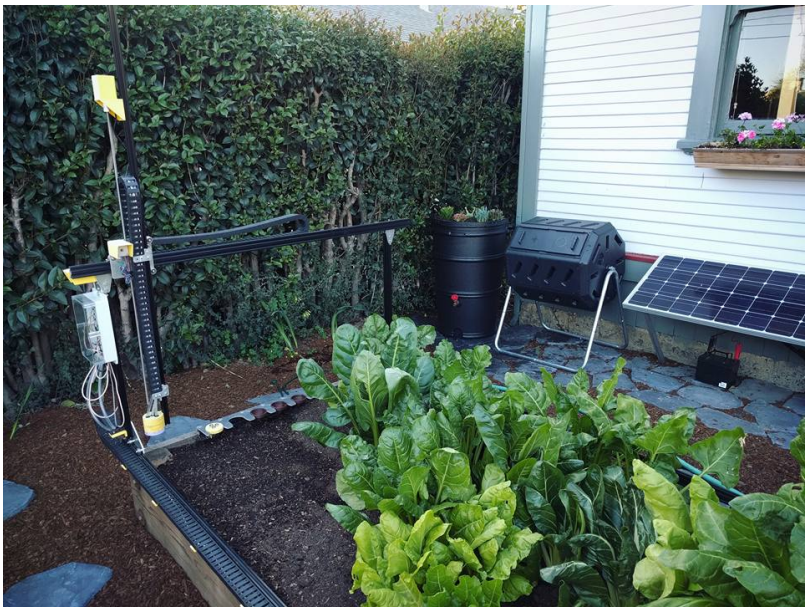


A szivattyúhoz egy kb. 1 méteres PVC csővel vezettem el a vizet, onnan pedig tovább a növényekhez, amit forrasztópákával kilyukasztottam 4 helyen, a 4 paradicsompalántának, amik így egyenletes lyukakon kapták a folyadékot. A csövet gyorskötözővel rögzítettem.

Mivel a képeken nem látszik annyira, hogy mit hogyan kötöttem össze, készítettem egy ábrát hozzá:



A projekt tehát végül így állt össze, összköltsége szinte elenyésző volt, egy komoly bolti növénygondozó rendszerhez képest, és bővítésre még rengeteg lehetőség van. Ha lesz rá energiám, szeretném bővíteni a rendszert, és 3D nyomtatóhoz hasonló beavatkozóval gondozni a növényeket, mint ahogy az ábrán látható.



Maradék képeket, videókat a dokumentum mellé csatolom.

LINKEK:
indexes hírek:
http://index.hu/tudomany/egeszseg/2016/05/19/nem_is_veszelyesebb_a_genmodositott_elelmiszer/
http://index.hu/nagykep/2015/04/15/the_third_day_monokultura_mezogazdasag_sivarsag_genseb eszet_uveghaz/
robotok:
https://www.youtube.com/watch?v=JZC77oN252g#t=111
https://www.youtube.com/watch?v=uNkADHZStDE
pythonosok:
http://stackoverflow.com/questions/30071886/how-to-get-current-time-in-python-and-break-up-into-year-month-day-hour-minu
http://stackoverflow.com/questions/9752958/how-can-i-return-two-values-from-a-function-in-python
http://stackoverflow.com/questions/676172/full-examples-of-using-pyserial-package
http://pvbookmarks.readthedocs.io/en/latest/devel/communication/serial/python/examples/pyserial/
növénytermesztés:
http://www.tankonyvtar.hu/hu/tartalom/tamop412A/2011-0085_precizios_novenytermesztes/ch01.html