DJI Drone Developing

① Motivation :

① Advisor's project

② Extend the control range of DJI Drone → original 2 km.

③ Extend ability of the controllen

④ 7 days from scratch. (6 people)

2. Functionality

① Use 3G/4G to control drone with Rasberry Pie

② Manipulate the flight

③ Can control Drone from a long Distance

④ Use BaiDu map SDK → show Drone's place

3. Techs Used & Implementation.

① Socket can't be in main thread ⇒ Use Handlon to send data to main thread.

② Client-server Arciteechare

③ used DJI onboard SDK

④ used " $ $ ---$" package

4. Road Blocks

① Wanted to use Poin 2 Point Arch → can't get drone's IP Address

② Latency is huge, in terms of controlig the Drone

③ Test is hard, Beijig is politcal Center, doesn't allow us to use Drones.

④ Java socket ⟷ C++ socket ⇒ byte Stream

⑤ Design the data package structure → team conflict → add package length

↳ sticky package → regex

⑥ Send-receiving Rate: Drone 10ms. mobile 100ms

Slides Controller.

1. Mottivation

① Many students don't have a remote controller when doing presentation

② Some times you need remote touch pad when doing presentation

③ IP free

2. Functionality

① Connect your computer to mobile hotsput → you are good to go

② Use touch screen or volumn button to go next slides or previous slides

③ Remote touch pad

④ Laser pen effect → sensors

- ① point your mobile to the center of the screen
- ② Set the central point.
- ③ Accelarotor, gravity, gyro, orientation ⇒ delta change ⇒ move the cursor.

→ Now this could be done by AR ?

3. Techs used & Implementation

① Java side → Robot class

② socket → UDP : Reason lost package can be lost. faster (Datagram socket,

③ AndroidManifest : Permission ; register activity

④ Sensor : SensorManager ; getSystemService(SENSOR_SERVICE)) registerListener ; onSensorchanged.

4. Dead Blocks.

① How to map (x,y) from small screen → big screen ⇒ By moving the delta distance

The distance moved related to its acceleration.

→ $\Delta y = \frac{1}{2} a t^2$ → (start to end touch)

$a = \frac{V_{sample}}{t_{sample}} \longrightarrow \frac{S_{20\,pixels}}{t_{sample}}$ ⇒ we could do better : dynamically sample a and s

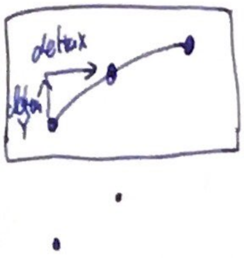
⇓

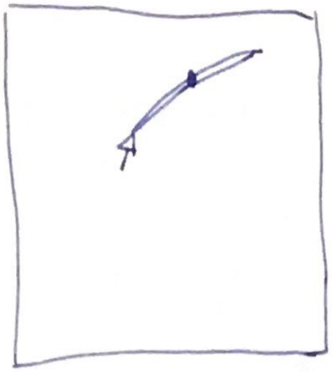
Hard to depict perfect curve.

→ Two dimensions.

② find IP Address : 192.168.43.255

③ Sensor → Hard to Manage





$\text{initX} + \Delta S_x \longrightarrow \text{finalX}$

$\text{initY} + \Delta S_y \longrightarrow \text{finalY}$

① when a is small

$\text{finalX} = \text{initX} + \underset{\curvearrowright}{\Delta S_x} \rightarrow 1$ pixel.

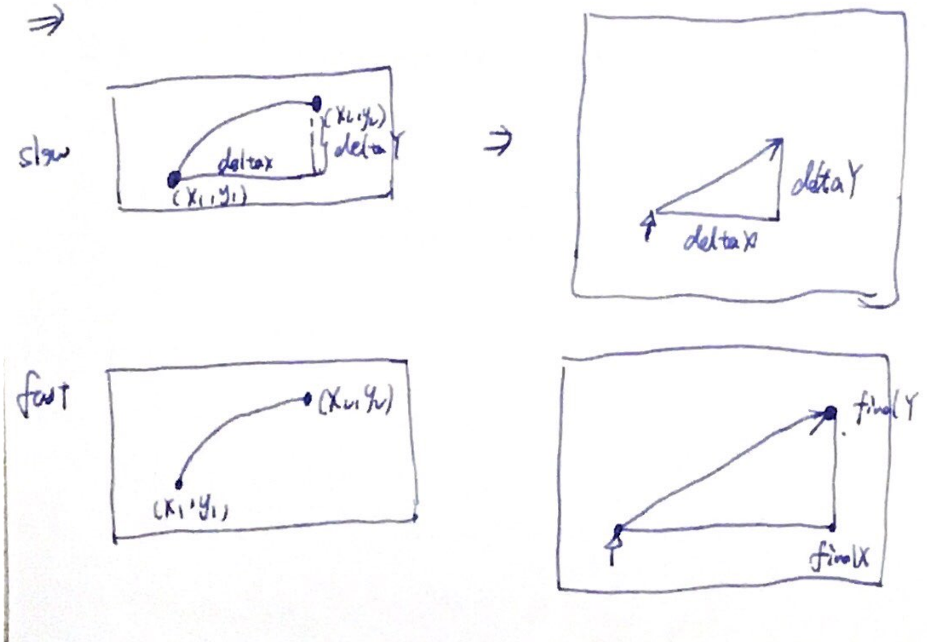
② when a is big.

$\text{finalX} = \text{initX} + \frac{1}{2} \underset{\longrightarrow}{a_x t^2}$ ~~$\frac{S}{t}$~~ $\frac{V}{t} \rightarrow V = \frac{S_x}{t}$

$\text{finalY} = \text{init} + \frac{1}{2} a_y t^2$

③ Actually depicting straight line from $(x_1, y_1) \rightarrow (x_2, y_2)$

⇒



☆ a and t is computed by front end → send to back end

↳ backend use a and t to compute finalXY

↳ send

(x, y, a, t)

(x, y, a_x, a_y, t)

if $a=0$ ⇒ go from Init → (x, y) with Δ

If $a>0$ ⇒ go from init → $\text{init} + \frac{1}{2}at^2$ with Δ

Candy Factory

1. Motivation:

① Teach kids fraction in a more intuitive way

② Kids can learn through playing, making learning [illegible] to be a fun thing.

③ Based on cloud → work under different platforms → different school uses different tablets or computers.

④ Teachers can collect data from their students, analyse performance and give suggestions.

2. Functionality:

① You got an order and candy bar. choose the right candy bar, slice it and concatenate to satify the order.

② Levels: level 1 → discreate candy ; level 2 → continuous candy
level 3 → nominator > denominator ; level 4. level 5 → still designing

③ Timer and score + achievement

④ Backend →
- User account creation & Log in
- Admin — — ———
- Admin check performance of users
- Password Encryption
- Email 2-factor Login
- cookies to remember usename & password.
- Basic charts → pie / bar

3. Techs used (JavaEE + CSS + HTML + JQuery) + Implementation

① Front end - Backend Interaction → JSON

② clock hand → CSS keyframe

③ Use session map to pass parameters between pages.

④ Trade-offs → when should front-end iteract with backend

⑤ mySQL + JSON

Road Blocks

① Code Refraction : extract consistent part out

② Game Design : when should front-end send data to backend.

③ Draw partial Candy : { CSS overflow = hidden. → use small div to hide whole candy.
Jquery append is synchronized, but under Chrome its not → add waiting time.

④ Long Js code vs. short Js code Trade offs.

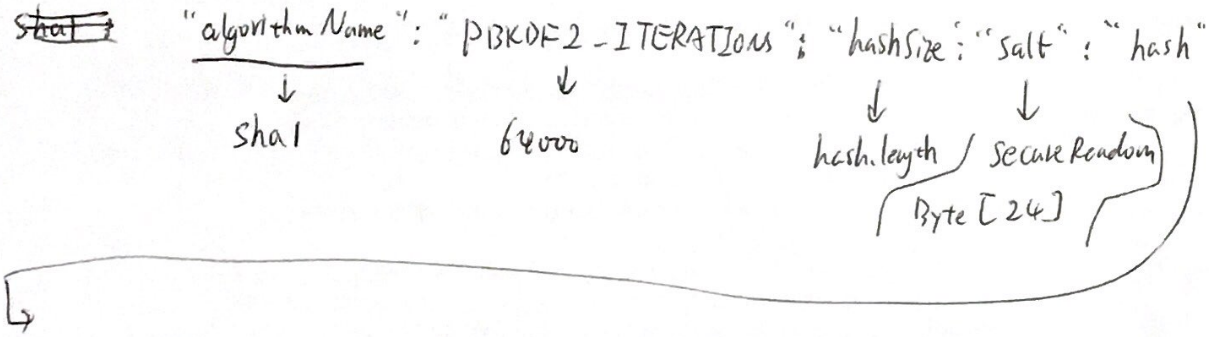
⑤ The naming of the variables in each page

⑥ if you use seperate shiftlog following each level, it doesn't reuse the code, but if you use one shiftLog, the parameters will be complex → granularity problem

⑦ Java script granuality

Password Encryption:

1. Encryption → Hash


~~Sha1~~
"algorithmName" : "PBKDF2_ITERATIONS" : "hashSize" : "salt" : "hash"
sha1
64000
hash.length
SecureRandom
Byte [24]

hash = pbkdf2(password, salt, PBKDF2_ITERATIONS, HASHBYTESIZE)

$\downarrow$ 18

Tracking Voter Turnout Web Application

1. Motivation

① Chose from project pool

② Analyse Voter Turnout Rate and reasons.

2. Functionality

① RSS News Feeding → RSS widgets.

② email ~~send~~ 2-factor login

③ play YouTube Video Related To Voter Turnout

④ Visualization with Charts + Filters.

⑤ List Voter Turnouts + Search Functionality

⑥ export as pdf ....

3. Techs & Implementations.

① Youtube → <iframe src = YouTube API />

② Encryptin AES — SHA1. → Hastle key (But should not).

③ email: Smtp + MiMe Message

④ Charts: Prime faces API → chart Model.

⑤ RSS Feeding: RSS widgets

4. Road Block

① Team work → lazy → schedule meeting and assign works

② Find Data → Import Into Database → Manually

③ Even don't login → can see other pages by url.

Supermarket Discount Recommand System.

1. Motivation.

① Eldors ~~tend to~~ loves supermarket discount, but the information is not managed or rendered. They missed lots of them.

2. Functionality.

① Normal user login

② Admin of supermarket login ⇒ "#Username"

③ Admin add items to correspondig supermarket

④ User brouse nearest suppermarket discount.

⑤ Recommand items

3. Techs & Implementation.

① DBUtil.getConnection() → Dao → ~~Bean~~ Domain → ~~Operation~~ service → service.java

Dao ↓ SQL

~~Bean~~ Domain ↓ objects. OOD

service.java ↓ servlet.java

② MyEclipse + MySQL

4. Road Block

① Design table

Admin

username   password   supermarket id.

Supermarket

~~name~~

Supermarket id   name   area

Items. Table name = supermarket Name.

User

username   pass   area . - - -

Linux Chatting Detail:

① 为了方便 debug, 我们在函数名前加了自己名字.

② 用 txt 文件 作为 数据库

③ multithread → pthread-create(&id, null, (void *) pthread_function , (int *) client_id);

④ chat → client +server Arch ; file transfor → p2p → server ↑↓ p2p client → client 2 .

JavaWeb Intern:

① sql connection → Driver Manager

② doGet + doPost

③ DBUtil . getConnection() → Dao → Bean → Operation ⇒ XXX Service . java

④ util {DBConnection → Dao → domain → service → servlet .

DJI:

① 粘包 → regex + 控制收发率

PPT Controler:

① map mobile → by Screen → { delta movement → accelerate↑, distance↑

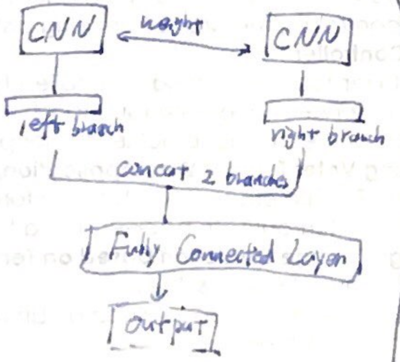
② sensor: { accelerometer, gravity, gyroscope, oriantation

Change Detection:

① robust → camera trembling , light changes , shadows

② Siamese network { input, input

③ improved Siamese network →



① Strong foundation of data which can support exciting projects. And I can solve the real-world problem using the data and resources. It feels great!

② Bloomberg has very strong impact in the world, it must be really cool expecting my team's project being landed.

③ Most of the people at Bloomberg are very talented , and I can learn a lot from Bloomberg and grow very fast.

Also I have engineering background and machine leaning background, I think in Bloomberg I can combine them together to make max profit

$S=\frac{1}{2}at^2$
↓
$\frac{v_t}{t}$

UI Chat Tool

1. Motivation

① Develop light-weight Linux Chat Tool with UI

② Improve collabrare in LAN

2. Functionality.

① one -one chat

② Group chat

③ P2P File transmission

④ UI → GTK

⑤ prevent duplicate login

⑥ any Linux computer can be tle server

3. Techs and Implementation.

① Database → txt file → light-weight → low dependency.

② Client _server Arch

③ multi thread → pthread

④ p2p file transmission.

i) Request for Ip address

ii) mark occupied

iii) file transmission

iv) mark unoccupied.

⑤ fwrite (struct → { username
socket Id

⑥ when user login

⇒ struct[ ]. add(user, socket id)

↳ notify all

when close tle window.

⇒ send signal

⇒ remove (user. socket id) ⇒ close it

⑦ pthread _create

(&id, NULL, (void*) pthread_function,

(int*) client_fd)

4. Road Blocks.

① Integrate UI code and backend code

② Arrange codes ⇒ messy since one functionality need many line of codes.

Change Detection

1. Motivation

① Traditional Method not robust enough to camera trembling / light changes / shadows and etc.

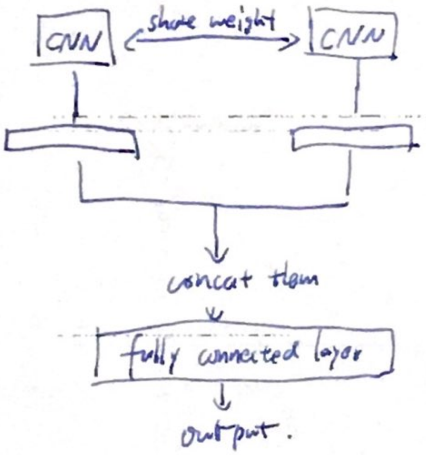
② ML can classify better

2. Functionality.

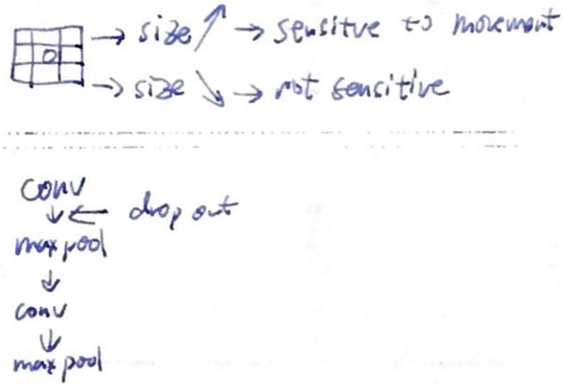
① Given 2 ~~series~~ samples of pics. give the change gray picture.

3. Tech used & Implementation

① Siamese network → 2 inputs.



CNN
share weight
CNN
concat them
fully connected layer
output.



→ size ↑ → sensitive to movement
→ size ↓ → not sensitive
conv
drop out
max pool
conv
max pool