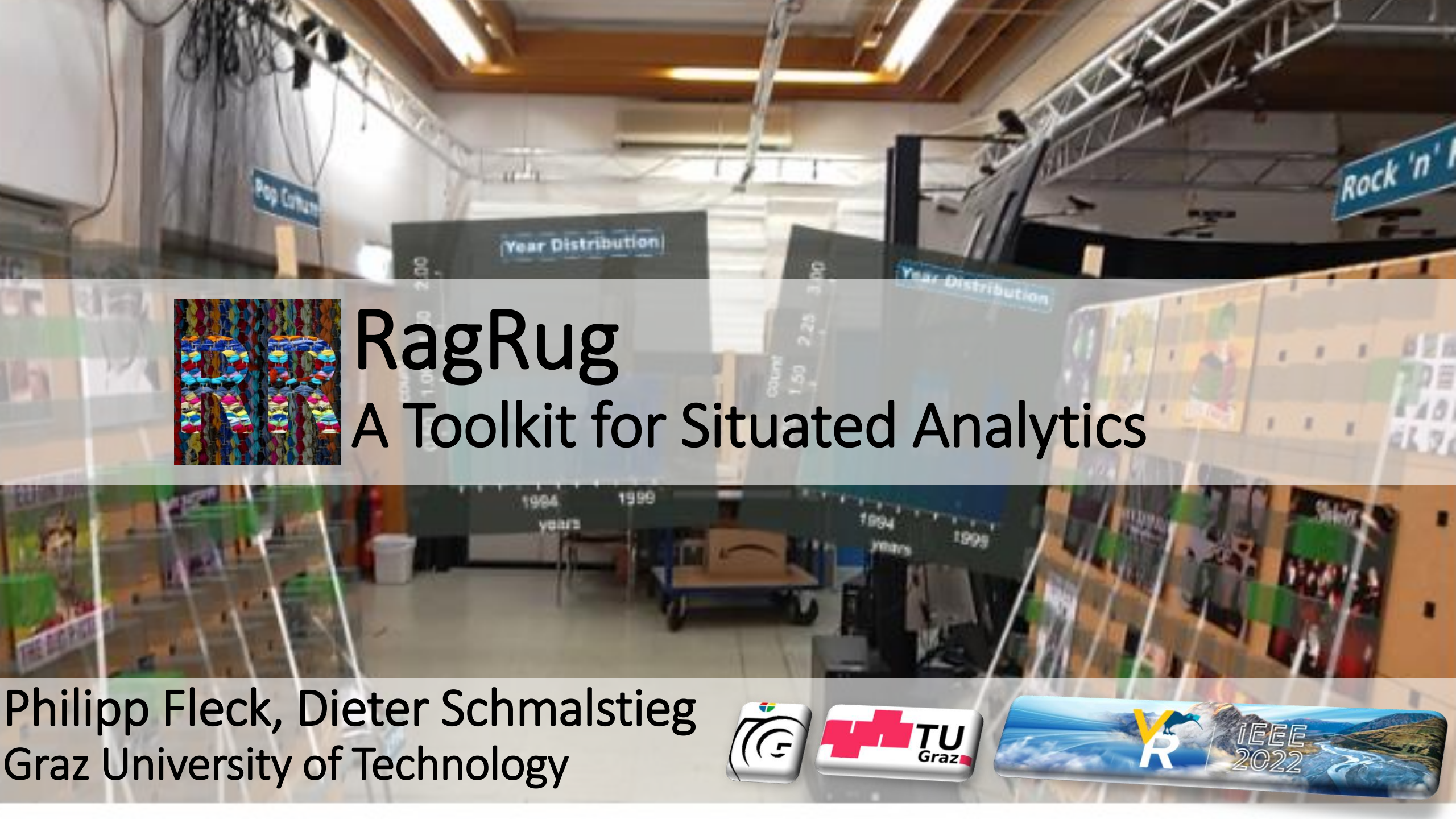


RagRug
A Toolkit for Situated Analytics

Philipp Fleck, Dieter Schmalstieg
Graz University of Technology

Links

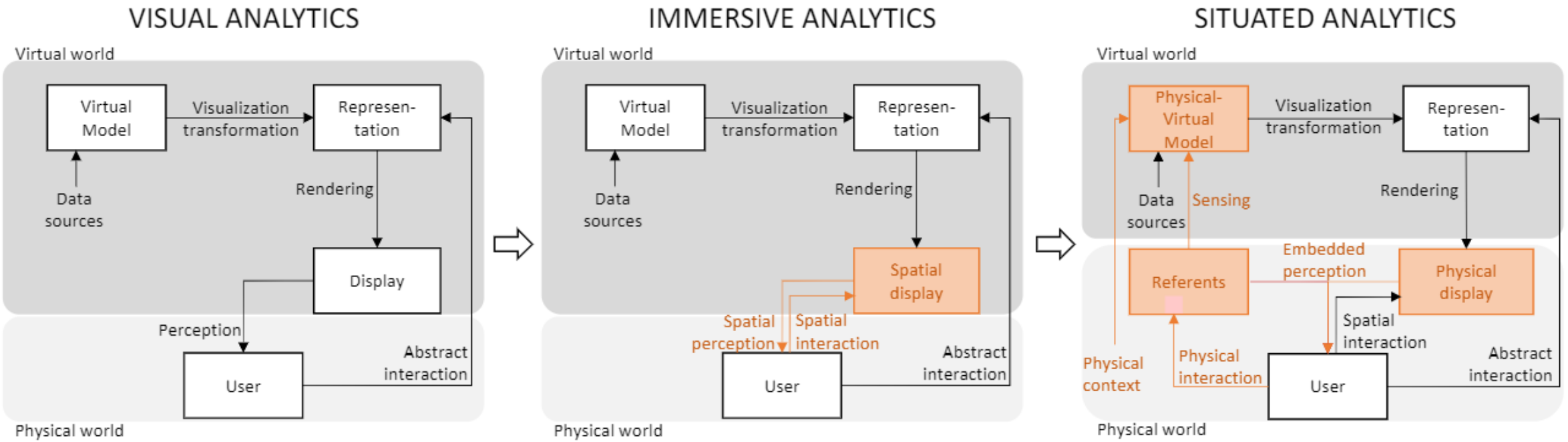
- Github
  - https://github.com/philfleck/ragrug
- Paper (**RagRug: A Toolkit for Situated Analytics**)
  - https://ieeexplore.ieee.org/document/9729627
- Videos
  - https://www.youtube.com/channel/UCCoAl_2xUJhpzH7vaEiLVmA
- Bibtex

@ARTICLE{9729627, author={Fleck, Philipp and Sousa Calepso, Aimee and Hubenschmid, Sebastian and Sedlmair, Michael and Schmalstieg, Dieter},
journal={IEEE Transactions on Visualization and Computer Graphics},
title={RagRug: A Toolkit for Situated Analytics}, year={2022}, volume={}, number={}, pages={1-1},
doi={10.1109/TVCG.2022.3157058}}

Situated Analytics

Situated Analytics

- Physical-Virtual Model
- Physical display
- Refferents (passive and active)



Visualization Toolkits

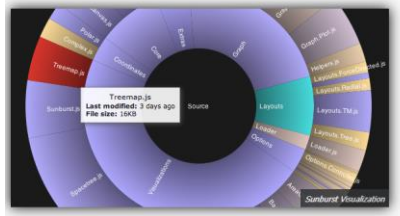
- D3, Vega, InfoVis Toolkit ➔ JS, Browser
- IATK, U2VIS, DXR ➔ 3D, Unity

Data-Driven Documents

JavaScript InfoVis Toolkit

Create Interactive Data Visualizations for the Web

Demos | Learn | Discuss | Contribute | Blog | About



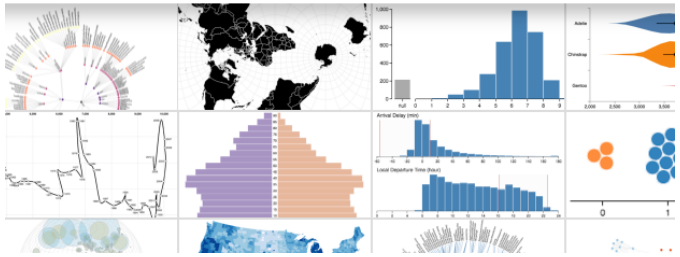
The JavaScript InfoVis Toolk provides tools for creating Interactive Data Visualizatio the Web.

Download

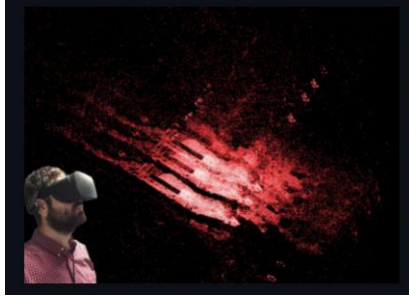
Current Release: 2.0.1

Donate

Vega – A Visualization Grammar



IATK 1.0 (Mala): Immersive Analytics Toolkit



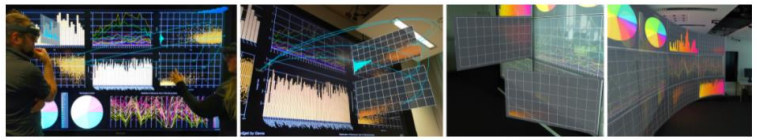
IATK: Immersive Analytics Toolki
you build **high quality, interactiv**
visualisations in **Immersive Envir**
(Virtual/Augmented Reality). Us
to create data visualisations inter
play and view and interact with y
simple code to use the *IATK* core
make your own interactive visuali

IATK is an ongoing and open pro
with features that we would like t
future.

Personal Augmented Reality for Information Visualization on Large Interactive Displays

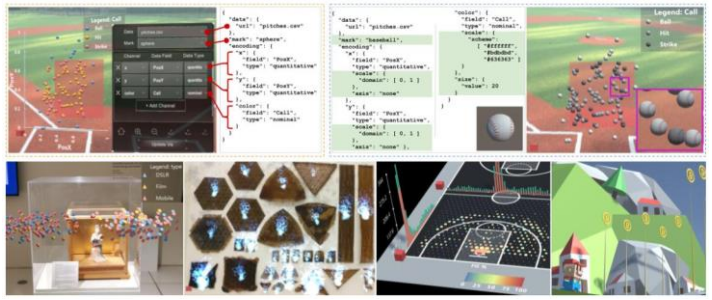
Patrick Reipschläger (M.Sc.), Tamara Flemisch (M.Sc.), Prof. Dr.-Ing. Raimund Dachselt

Leider ist dieser Inhalt nur in Englisch verfügbar. Aus praktischen Gründen wird der Inhalt unten in dieser Sprache angezeigt.



DXR: A Toolkit for Building Immersive Data Visualizations

Ronell Sicat, Jiabao Li, JunYoung Choi, Maxime Cordeil, Won-ki Jeong, Benjamin Bach and Hanspeter Pfister



Visualization Authoring

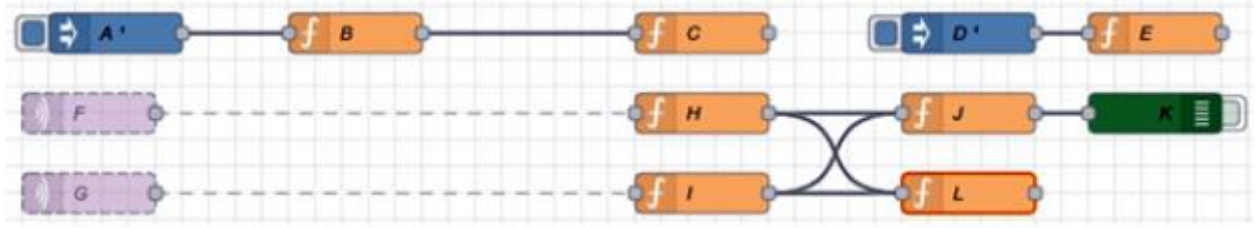
- Grammar of graphics for visualization spezification
- Instrumental Interaction
  - generalized form of direct manipulation
  - Interaction instruments combine physical and virtual parts
- Referents can serve as physical instruments in situated visualization, while the stages of the visualization pipeline can serve as virtual instruments
- we enable the combination of physical and virtual instruments in a completely open manner, using free-form dataflow.

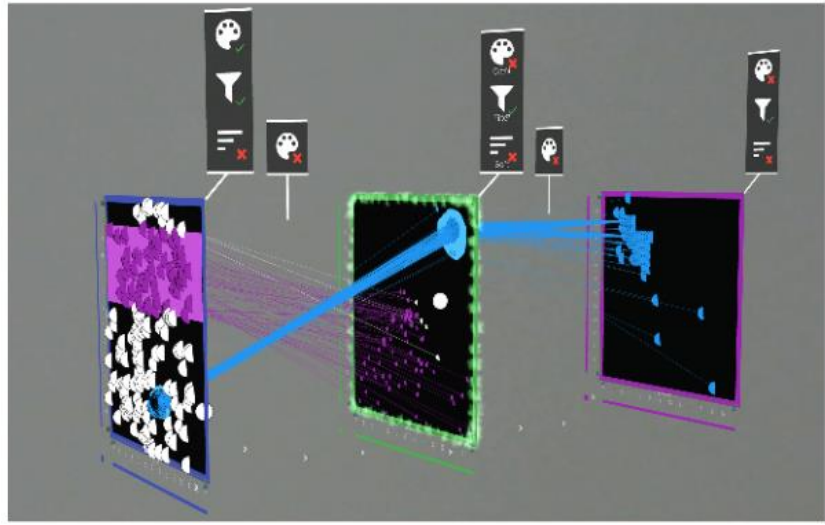
RagRug

https://github.com/philfleck/ragrug

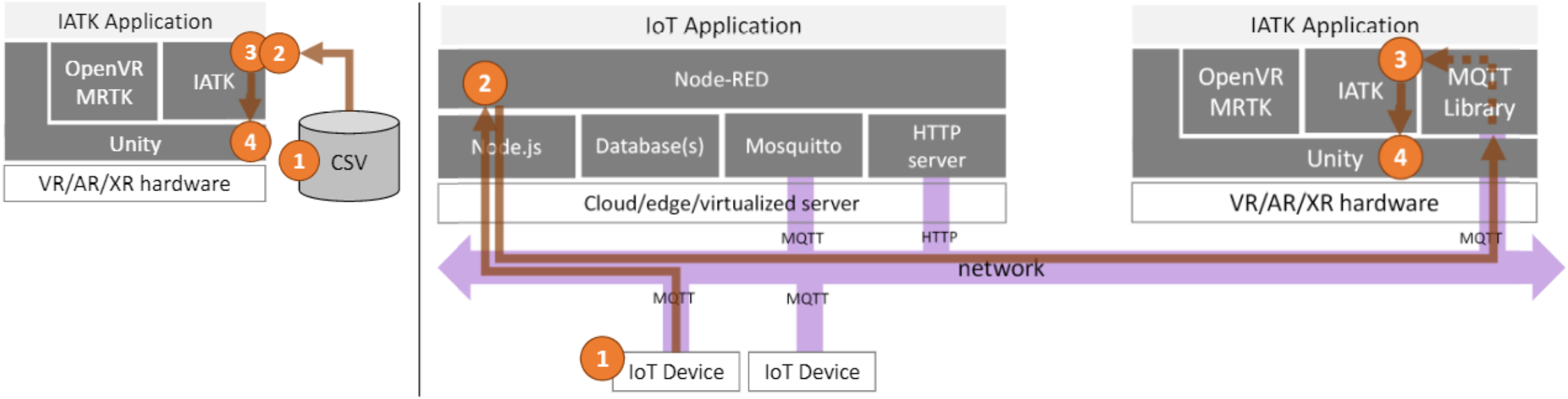
RagRug

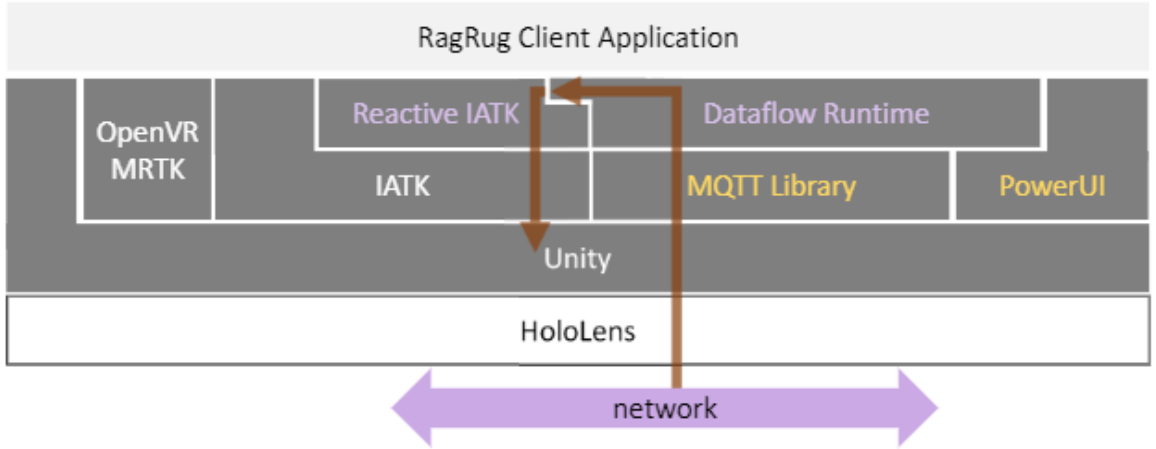
- Toolkit for Situtaed Analytics
  - Visual Analytics → Immersive Analytics(spatial) →Situated Analytics(referents + phys-virt-model)
- Physical-Virtual World Model
- Distributed Real-system
  - NodeRed, MQTT, CouchDb, InfluxDB, PostGres, Unity, IATK, MRTK, Vuforia, PowerUI
- ClientSide Flow Interpreter → NR visual programming





VR/AR/XR/IoT Applications





Aspect	IoT app.	IATK app.	RagRug
Language	**Javascript**	C#	**Javascript**
Visual encoding	no	**Vega-Lite**	**Vega-Lite**
3D rendering	no	**Unity**	**Unity**
Interaction	**reactive**	fixed	**reactive**
Passive referents	**via database**	no	**via database**
Active referents	**via MQTT**	no	**via MQTT**

Unified programming for situated authoring

- Public / Subscribe
  - ➔ lose client-client / client-server communication
- IATK + Real-time datasource
  - ➔ Reactive IATK
- Clientside JS
  - ➔ NodeRed Flow interpreter
- NodeRed Flow can run on server and client
  - ➔ unified programming
  - ➔ distributed data-flows

RagRug

A Toolkit for Situated Analytics

Philipp Fleck, Aimee Sousa Calepso, Sebastian Hubenschmid,
Michael Sedlmair and Dieter Schmalstieg

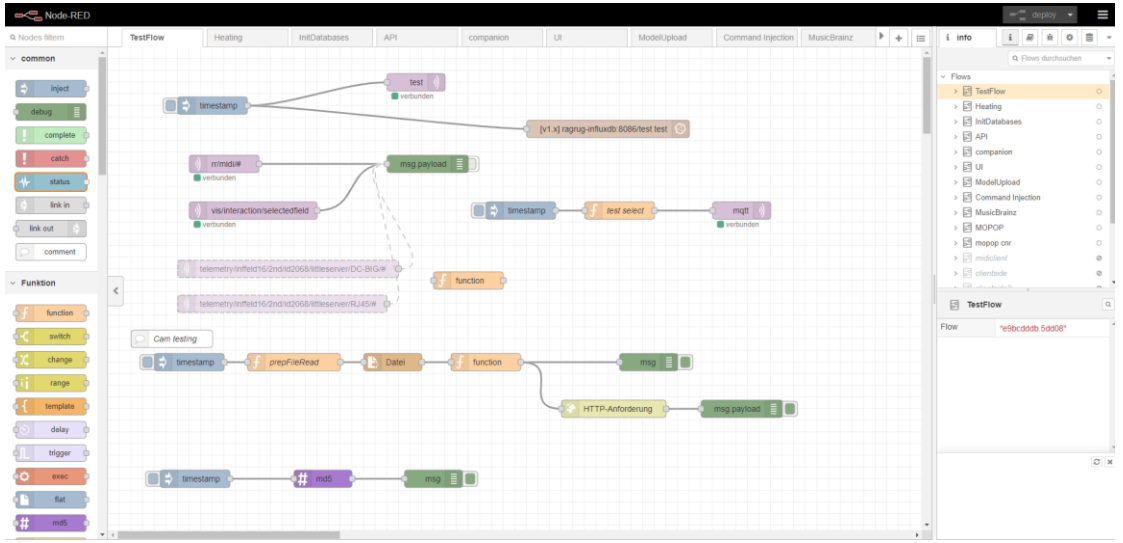
Examples

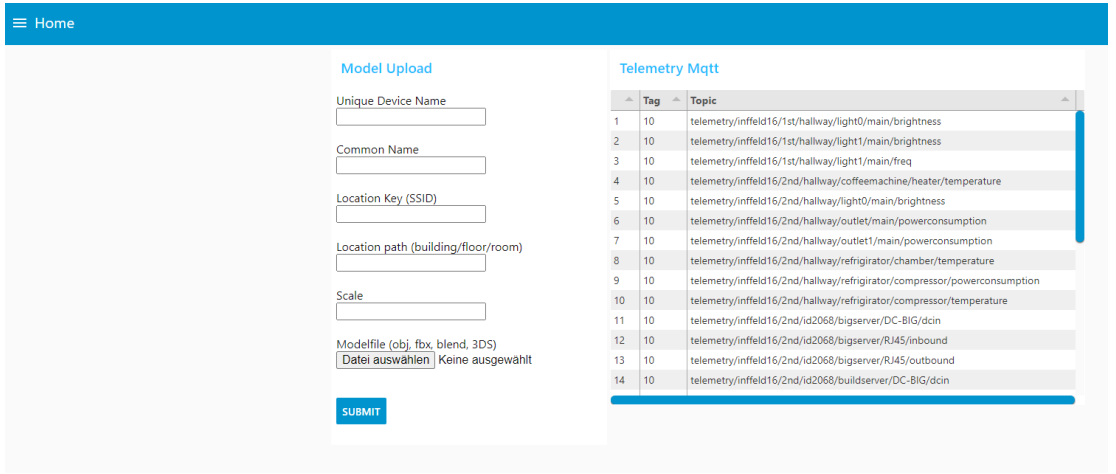
Royalty Free Music from Bensound

Graz University of Technology University of Stuttgart University of Konstanz

Node Red

- Orchestration backend
- Visual programming
- Often used for Home-Automation, IoT, device orchestration, prototyping backends but also in production
- Based on Node.js, backend, web-frontend
- A lot of pkg/function to install
- Support for many databases
- WebApi, Sockets, MQTT, much more





Modeling the world

Publish – Subscribe via MQTT

- Losely coupled → no hard links, decouped code
- Producer and Consumer do not other each other
- Broker in the middle for message routing
- Topic based with support for hierarchies and wildcards
  - telemety/sensors/#, telemetry/sensors/+/temperature
- Allows for easy data discovery and notification
- Use cases:
  - Input to an visualization, broadcasting of data (Sensors, HMD-pose), …

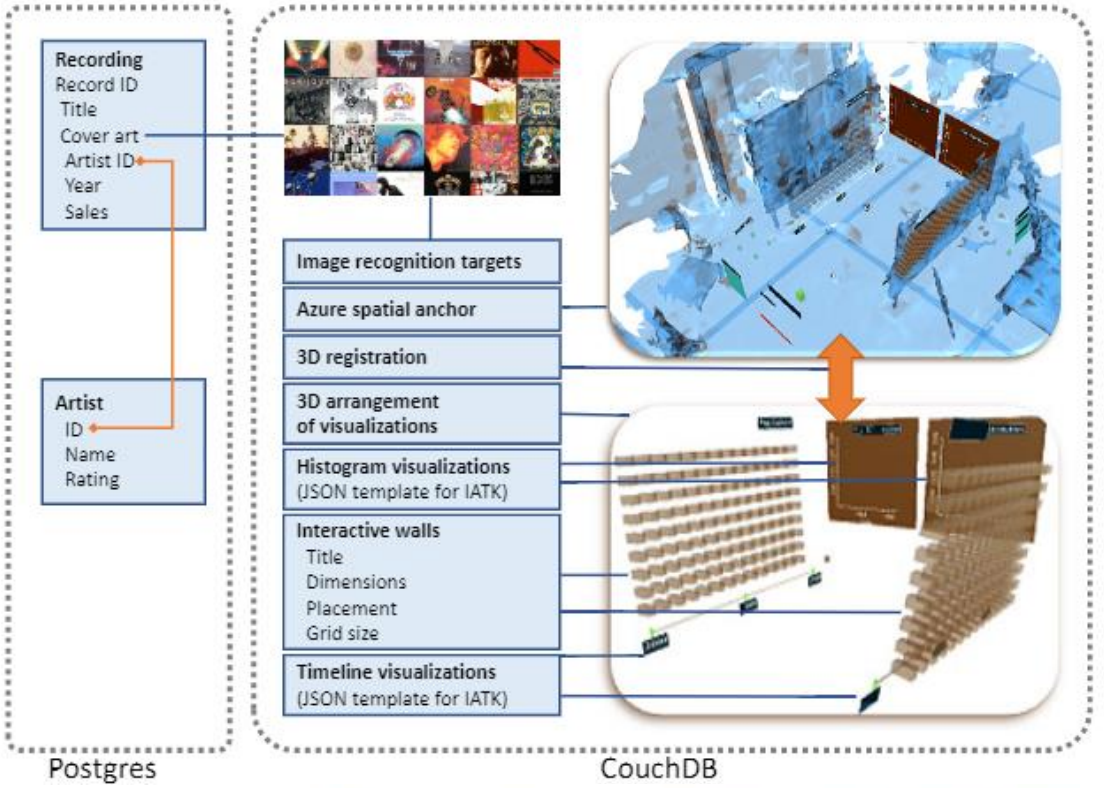
Locations

1. We are within a Building
2. On a Floor
3. Within a Room
4. Where Devices are
5. Which have Parts
6. Where Parts have Abilities
7. Which produce Data

- Localize within a room using its SSID/BSSID
- MQTT
  - Messaging protocoll
  - Publish/Subscribe based on topics
  - Topics follow Location concept
  - Data-prefix: *telemetry*
  - telemetry/building/floor/room/device/part/ability

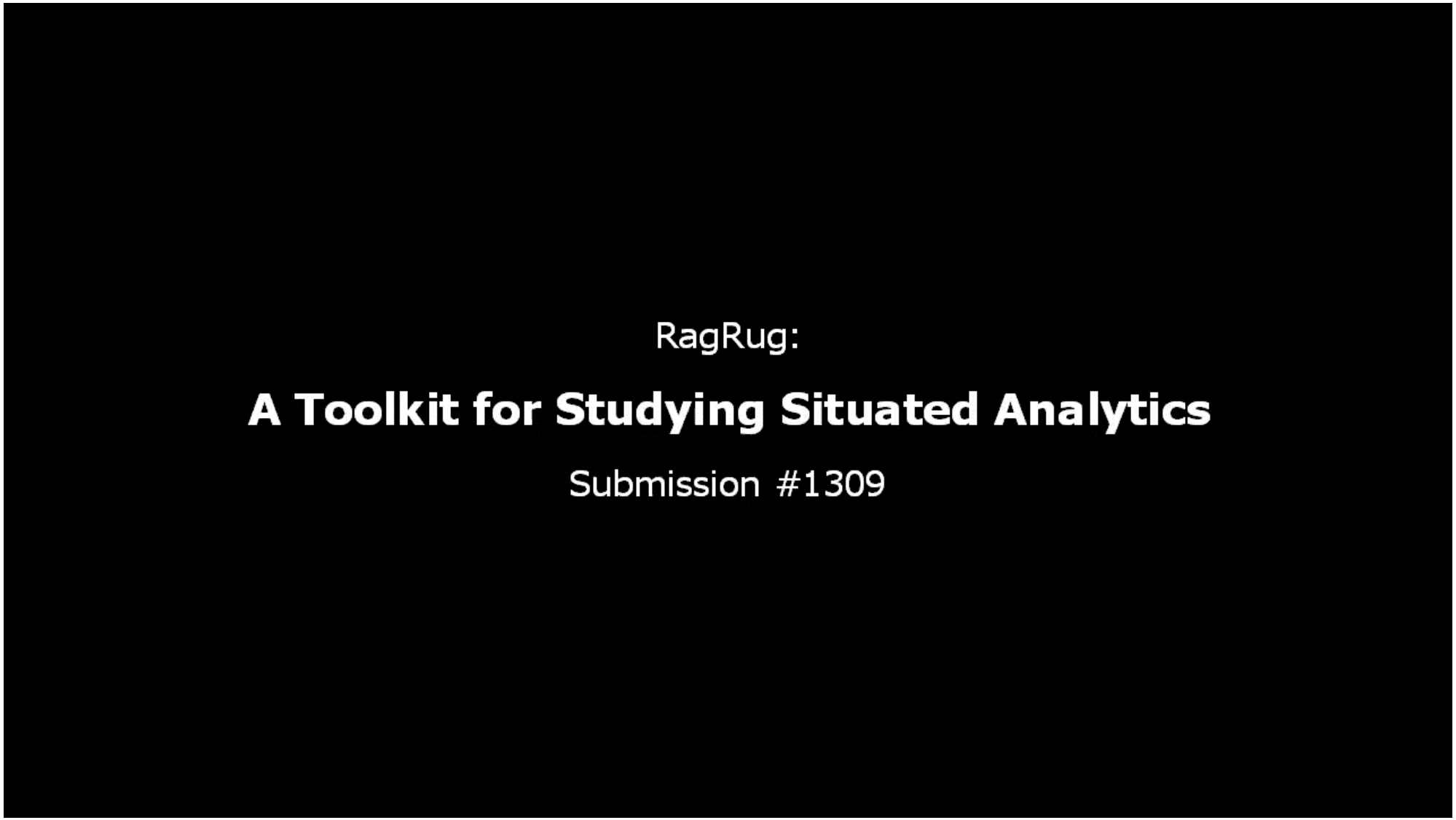
Digital twins

- Situated analytics
  - No (or little) toolkits out there
  - Hard to do (implementation havey)
  - Hard for visualization people to explore



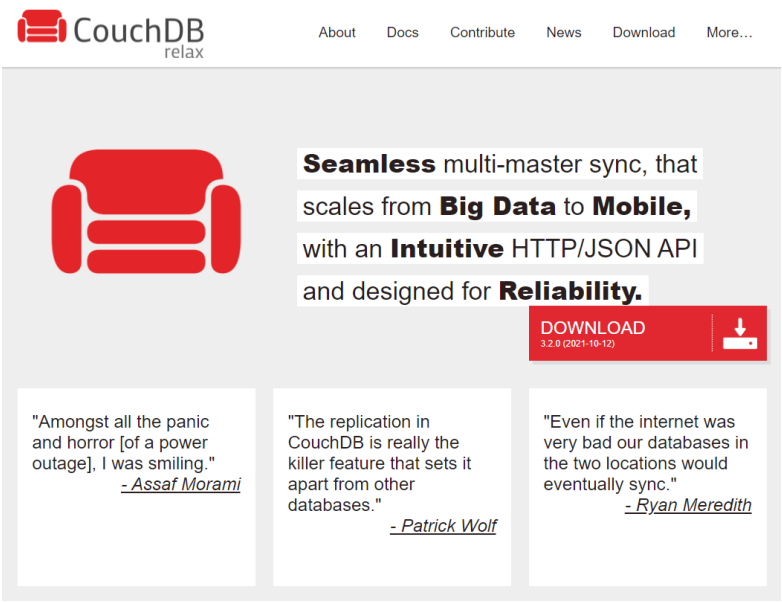
- Showing benefits of (passive or active) referents against a more traditional way
- Little comparison or user studies done

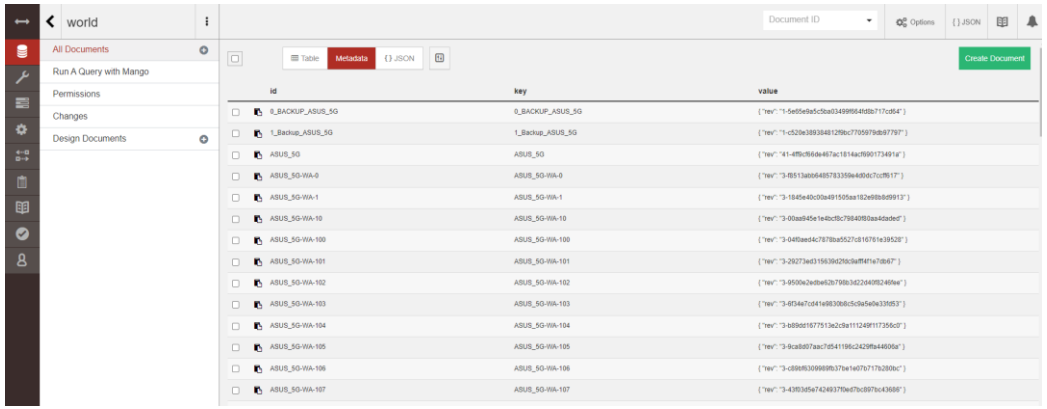
Mopop 1.0


RagRug:
A Toolkit for Studying Situated Analytics
Submission #1309

CouchDB – Stores the World

- Scalable, fast, document based
- Replication
- Web-ready, json based
- Rest-API per default
- Support views, map-reduce tasks
- Large documents, attachements
- Non - Relational



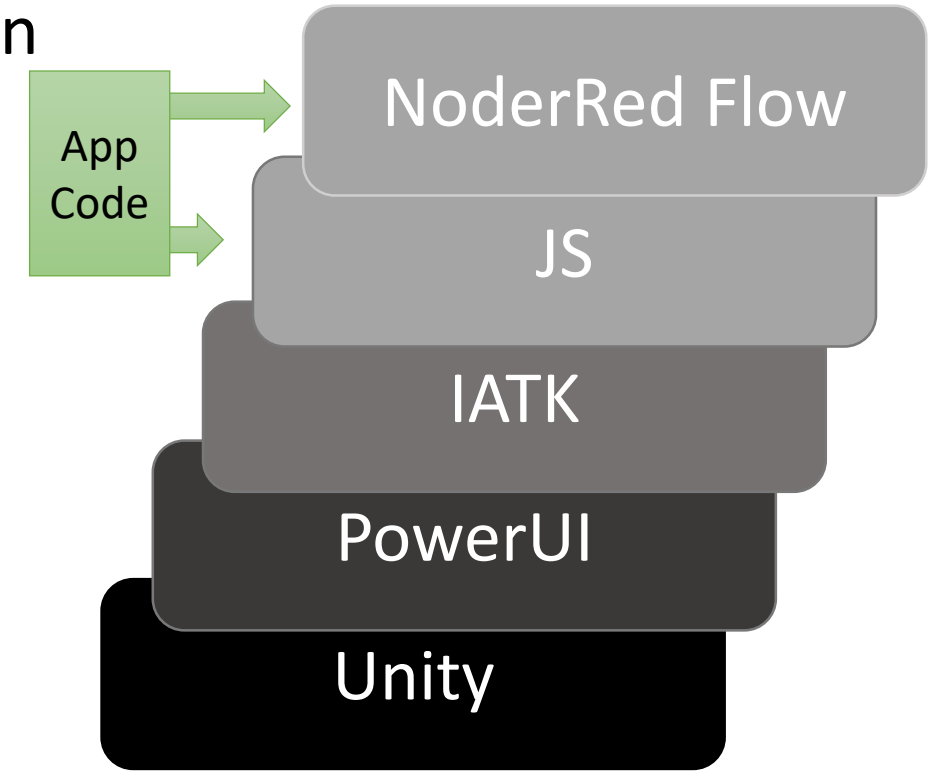


AR Client

ARClient

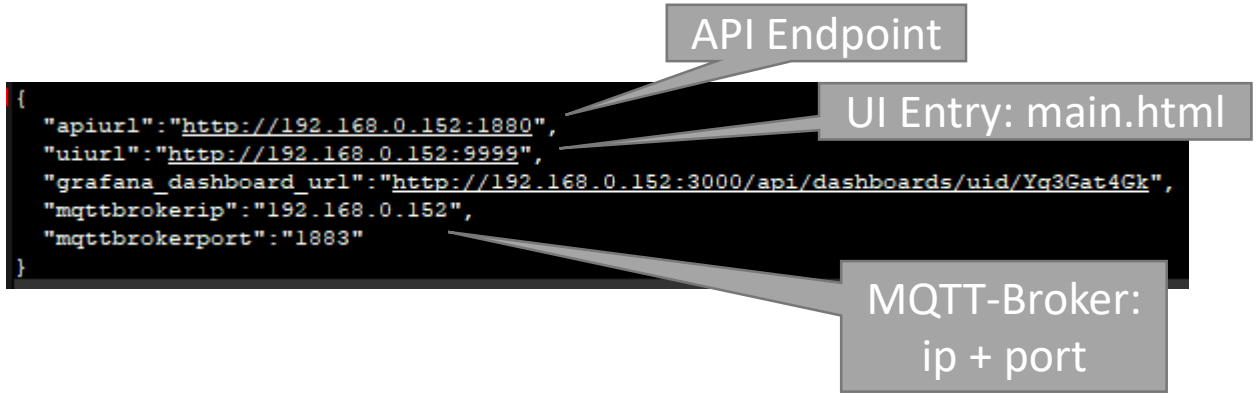
- The cnr.js (flow interpreter) needs JS to run
- JS is the application layer
- IATK for the Visualization
- PowerUI for JS (JIT) and C# bindings
- Unity for rendering and device access

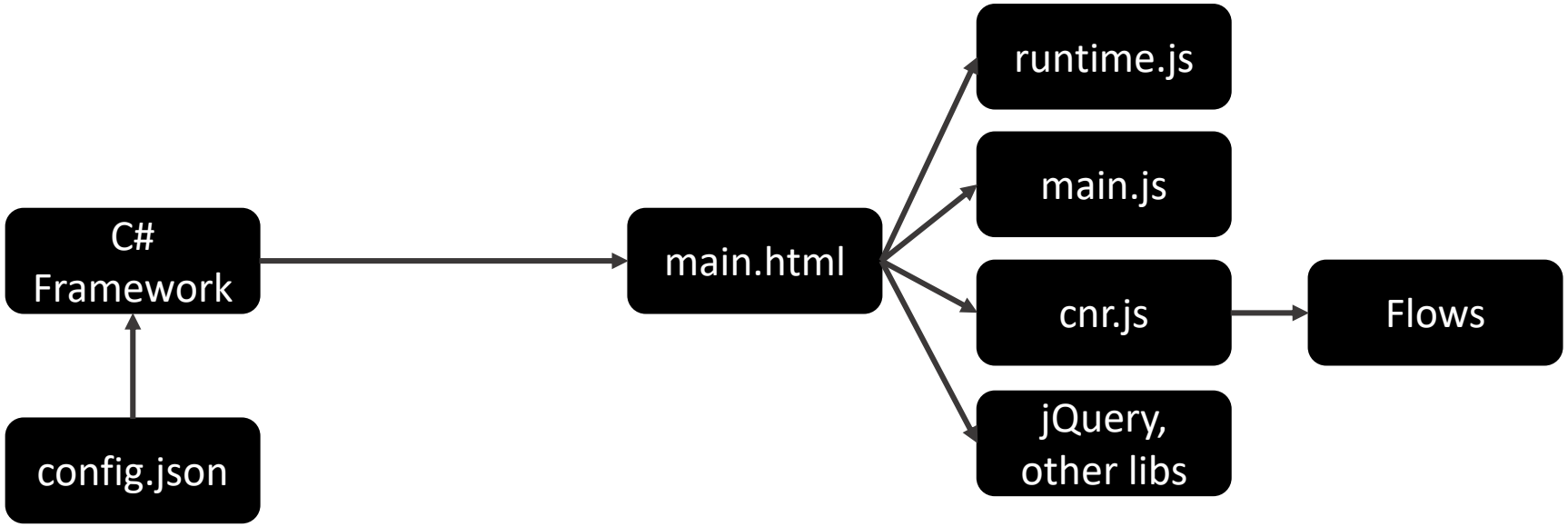
- Following web-style principles: Fuzzy loading, Lazy loading, hot-plugging
- No compilation



ARClient: Default Loading Procedure

- Starts from config.json
- Loads on default main.html
- Using RRs main+runtime:





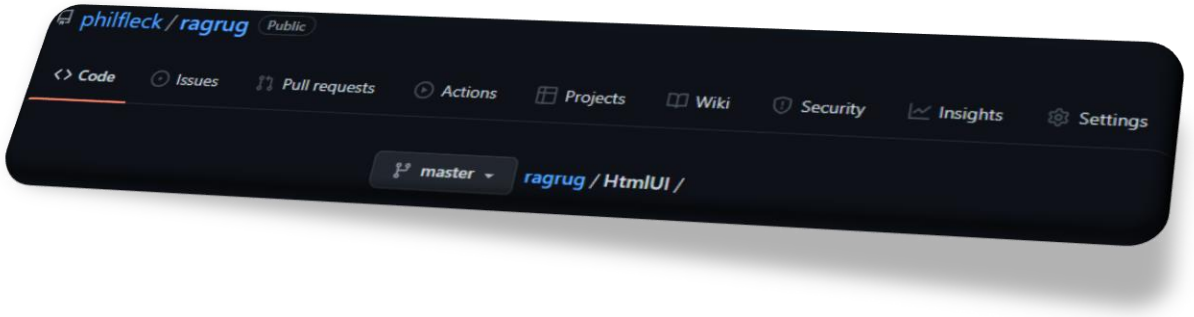
ARClient: runtime.js

- Functions like RT.VIS.GetNewChartObjectIATK2 are implemented in the ARClients Runtime
- Hotloaded from the backend at startup (default behavior).
- The runtime provides helper functions for common tasks such as
  - Creating an empty visualization. By providing a higher level wrap we can make IATK function easily accissible and referenceable without ommiting its default API, which still can be (and is) used to certain tasks.
  - Unity interaction
  - File I/O, device related functionality
  - Access to C# Runtime.cs, which is the counterpart on C# level
  - Easy access to the MQTT layer
- Defines common object structures
- Implemets MACRO like functions (shortcuts)
- Access to special unity functions

ARClient: main.js

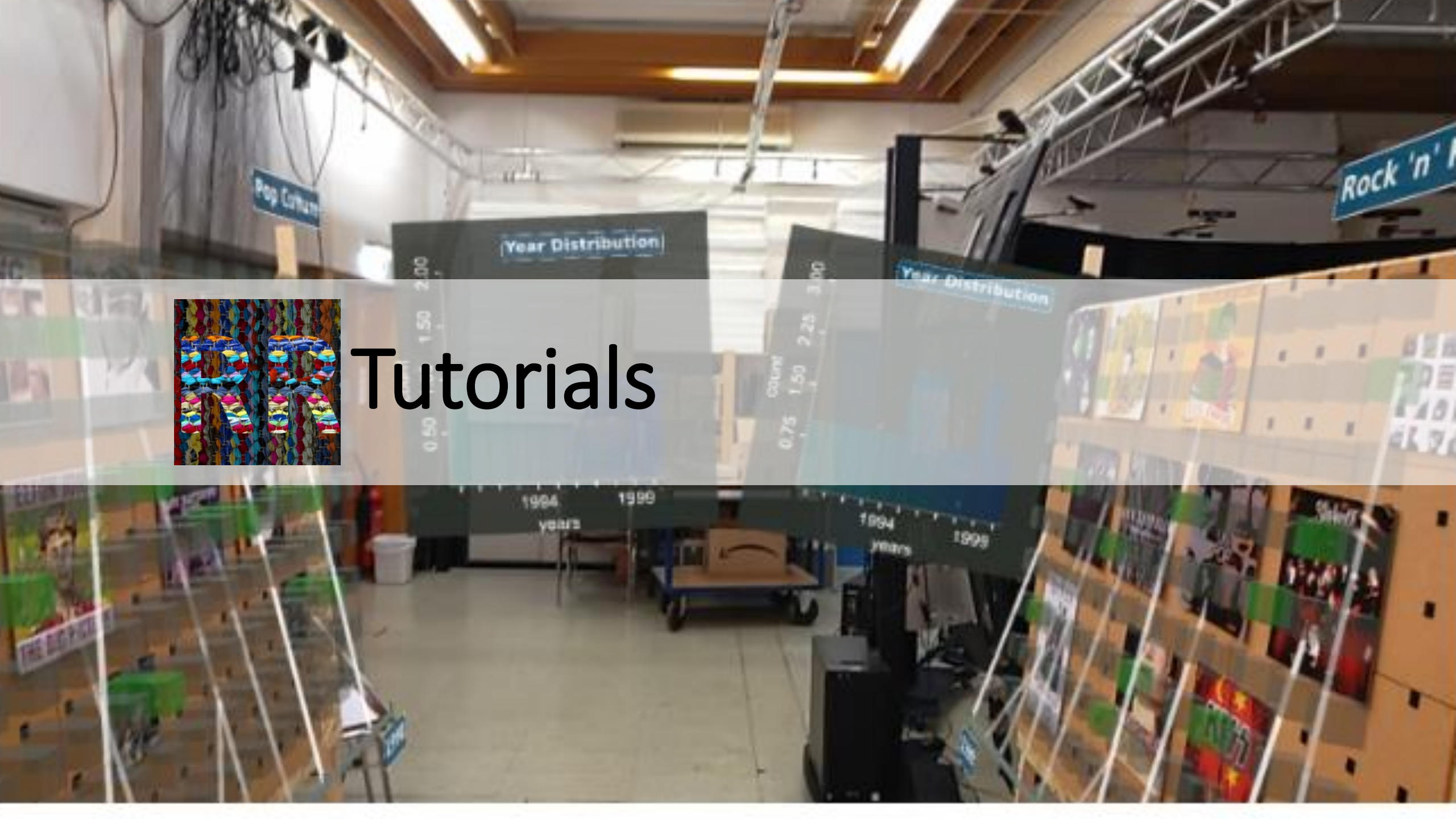
- Default application
- Offering sample UI and basic interactions
- Can load the World and instantiate devices
- Can load AssetBundles
- Showcase of default functionality like:
  - Location, Devices, Visualizations, Tracking, clientside-flows

ARClient: cnr.js



- Only deployed flows can be loaded (by ID)
- When active they will run on the server as well (client can execute disabled flow)
- Flow restriction by checking if run on client (property present, try-catch blocks)
- Support for multiple instances or context (1-to-many flow per DOM)
- Any js-file can load and instantiate the flow through cnr.js
- Growing support for nodes (currently: inject, MQTT-In, MQTT-Out, Function, Debug)


```
// cient filter
try {
    var n = MAIN.usercanvas;
    return msg;
} catch(err) {
    return null;
}
return null;
```


Pop Culture
Year Distribution
Rock 'n'
Year Distribution
count
years
1994
1999
years
1994
1999
Shirts

Tutorials

Tutorial overview

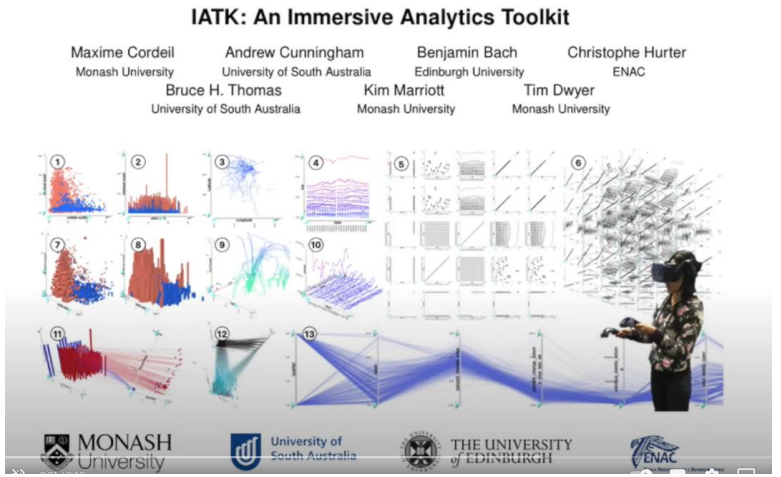
Tutorial	Visualization	Data Flow	Location	Tracking
IATK	X			
Smart Fridge	X	X		
Location			X	
Basement	X	X	X	
Target Tracking			X	X
ControllAR	X	X	X	X

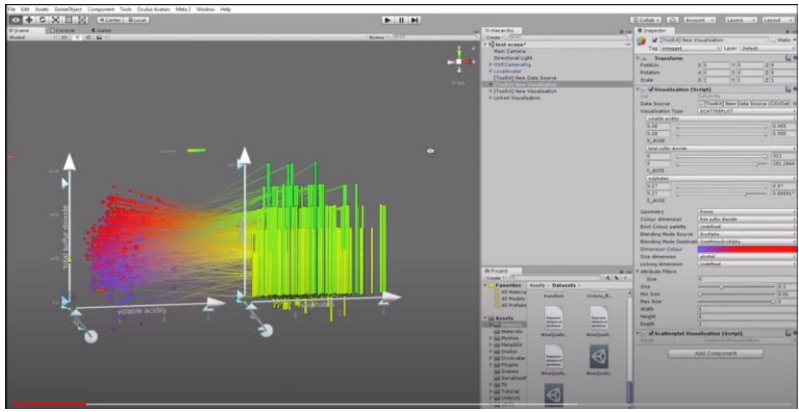
IATK Introduction

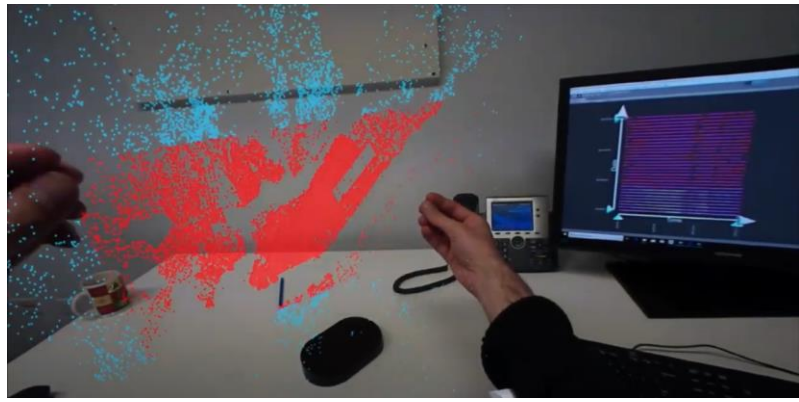
IATK Basics

- Unity prefabs
- Static data
- Setup within the Unity editor
- Uses internally JSON format for serialization
- Desktop based 2D gui
- Code based via SDK/API
- Support for immersive AR / VR

https://github.com/MaximeCordeil/IATK
https://www.youtube.com/watch?v=qiWIPePTsOA

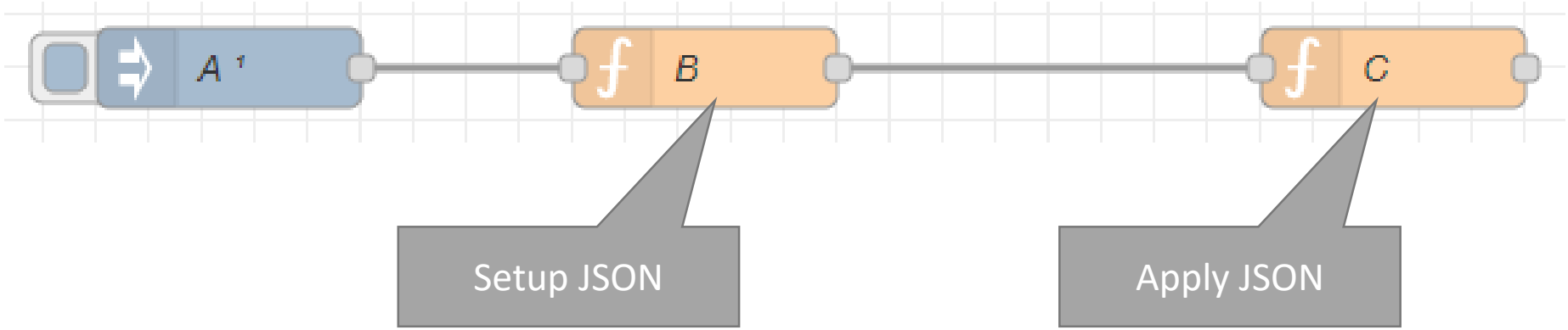


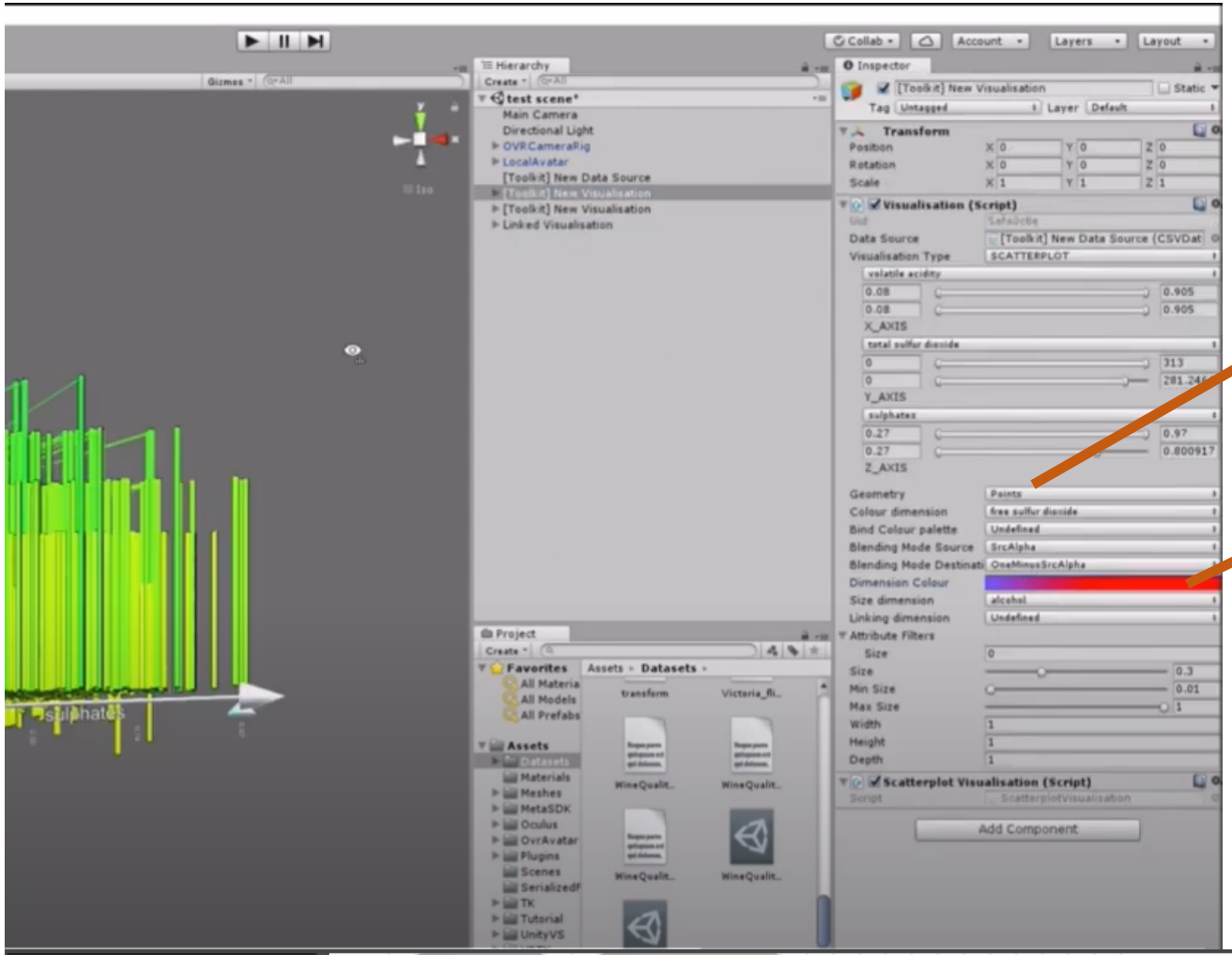




IATK in RagRug

- A target is to author visualization outiside the Editor and outside the C# world
- We can use js-c# bindings to directly write IATK-SDK code in js
- We leverage the IATKs JSON format to create visualizations within a NodeRed Flow
  - Pepared helper functions within runtime.js





VS

```
// B
var visData =
{
    "UID":"",
    "VisualisationType":"SCATTERPLOT",
    "parallelCoordinatesDimensionsAttributes":[],
    "parallelCoordinatesDimensionsMinFilter":[],
    "parallelCoordinatesDimensionsMaxFilter":[],
    "parallelCoordinatesDimensionsMinScale":[],
    "parallelCoordinatesDimensionsMaxScale":[],
    "Geometry":4,
    "AxiesKeys":[0,1],
    "AxiesValues":["id","temperature"],
    "ColourDimension":"temperature",
    "SizeDimension":"Undefined",
    "LinkingDimension":"names",
    "colourKeys":
    {
        "serializedVersion":"2",
        "key0":{"r":1.0,"g":0.0,"b":0.0,"a":1.0},
        "key1":{"r":0.0,"g":1.0,"b":0.0,"a":0.5},
        "key2":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key3":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key4":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key5":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key6":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key7":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "ctime0":0,"ctime1":65535,"ctime2":0,"ctime3":0,"ctime4":0,"ctime5":0,"ctime6":0,"ctime7":0,
        "atime0":0,"atime1":65535,"atime2":0,"atime3":0,"atime4":0,"atime5":0,"atime6":0,"atime7":0,
        "m_Mode":0,
        "m_NumColorKeys":2,
        "m_NumAlphaKeys":2
    },
    "colour":{"r":0.0,"g":0.0,"b":1.0,"a":0.5},
    "Size":0.2,
    "MinSize":0.04,
    "MaxSize":0.2,
    "VisualisationWidth":0.3,
    "VisualisationHeight":0.3,
    "VisualisationDepth":0.3
};
msg.payload = visData;
```

Geometry: Lines, Bars, Dots, …

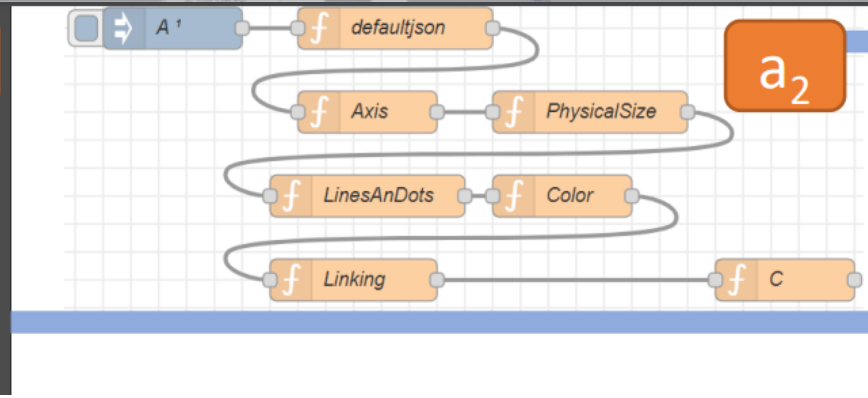
Dimensions

Color Keys

Visualization size

Physical size

a_1

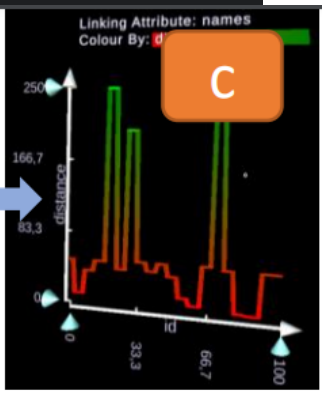


a_2

alternative

```
var visData =
{
    "UID":"",
    "VisualisationType":"SCATTERPL
    "parallelCoordinatesDimensions
    "parallelCoordinatesDimensionsMinFilter":[],
    "parallelCoordinatesDimensionsMaxFilter":[],
    "parallelCoordinatesDimensionsMinScale":[],
    "parallelCoordinatesDimensionsMaxScale":[],
    "Geometry":4,
    "AxiesKeys":[0,1],
    "AxiesValues":["id","temperature"],
    "ColourDimension":"temperature",
    "SizeDimension":"Undefined",
    "LinkingDimension":"names",
    "colourKeys":
    {
```

b



c

Smart Fridge

Tutorial

Overview

Tutorial	Visualization	Data Flow	Location	Tracking
IATK	X			
Smart Fridge	X	X		
Location			X	
Basement	X	X	X	
Target Tracking			X	X
ControllAR	X	X	X	X

Smart Fridge

- Consider the following scenario: The fridge door often remains open. Therefore we would like to setup an alarm and to record the actual temperature drop
- To obtain real values in real-time we install one of our nodeMCU based (ESP8266) sensors equiped with an temperature and and an ultrasonic distance sensor within the fridge. The distance sensor points towards the door, allowing to easily detect when the door is open. Measurements below 35cm indicate a closed door and above an open door.

Smart Fridge - Requirements

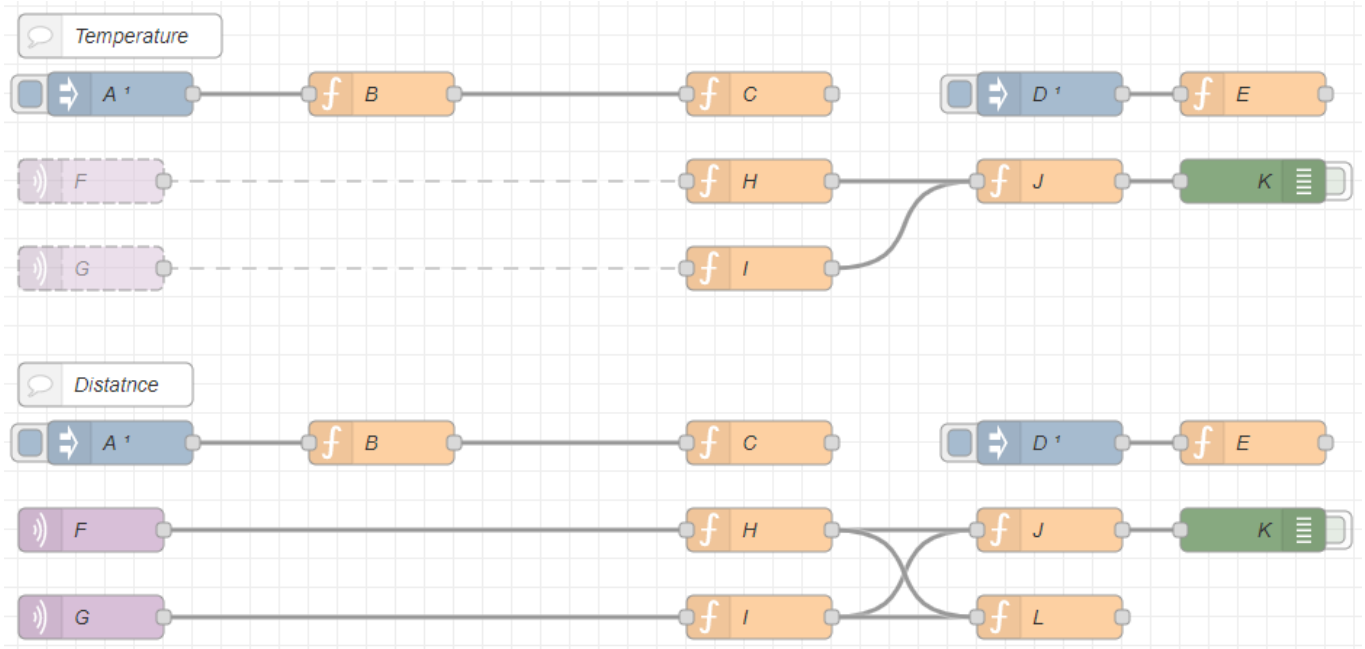
- Within the Hololens we would like to see two line-based visualizations as time series
- Temperature curve
  - X-Axis: running time series
  - Y-Axis: measured temperature
  - Z-Axis: disabled
- Distance curve
  - X-Axis: running time series
  - Y-Axis: measured distance
  - Z-Axis: disabled

Smart Fridge - Requirements

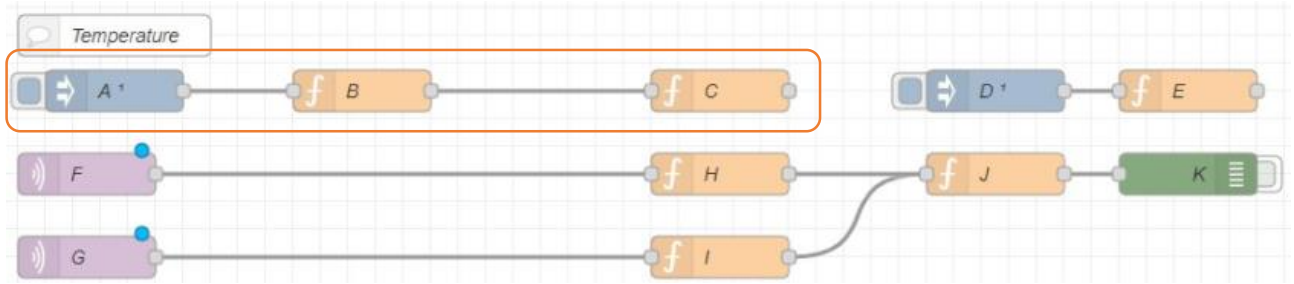
- An alarm should be played one the door is opened
- Stopped when closed
- Additionally a visual alarm should be visualized within he users field of view.

Smart Fridge - Implementation

- We can access NodeRed running within our backend (usually) on http://localhost:1880 and add a new flow.
- Load flow by ID url (eg http://localhost:1880/#flow/f3784d00.40ef8 -> f3784d00.40ef8)
- A different way to load a flow on the client is to define a fixed entrypoint for flows and to continue loading from there.



Smart Fridge




```
// B
var visData =
{
    "UID":"",
    "VisualisationType":"SCATTERPLOT",
    "parallelCoordinatesDimensionsAttributes":[],
    "parallelCoordinatesDimensionsMinFilter":[],
    "parallelCoordinatesDimensionsMaxFilter":[],
    "parallelCoordinatesDimensionsMinScale":[],
    "parallelCoordinatesDimensionsMaxScale":[],
    "Geometry":4,
    "AxiesKeys":[0,1],
    "AxiesValues":["id","temperature"],
    "ColourDimension":"temperature",
    "SizeDimension":"Undefined",
    "LinkingDimension":"names",
    "colourKeys":
    {
        "serializedVersion":"2",
        "key0":{"r":1.0,"g":0.0,"b":0.0,"a":1.0},
        "key1":{"r":0.0,"g":1.0,"b":0.0,"a":0.5},
        "key2":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key3":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key4":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key5":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key6":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "key7":{"r":0.0,"g":0.0,"b":0.0,"a":0.0},
        "ctime0":0,"ctime1":65535,"ctime2":0,"ctime3":0,"ctime4":0,"ctime5":0,"ctime6":0,"ctime7":0,
        "atime0":0,"atime1":65535,"atime2":0,"atime3":0,"atime4":0,"atime5":0,"atime6":0,"atime7":0,
        "m_Mode":0,
        "m_NumColorKeys":2,
        "m_NumAlphaKeys":2
    },
    "colour":{"r":0.0,"g":0.0,"b":1.0,"a":0.5},
    "Size":0.2,
    "MinSize":0.04,
    "MaxSize":0.2,
    "VisualisationWidth":0.3,
    "VisualisationHeight":0.3,
    "VisualisationDepth":0.3
};
msg.payload = visData;
return msg;
```

Geometry: Lines, Bars, Dots, …

Dimensions

Color Keys

Visualization size

Physical size

Create blank IATK Visualization using RR.Runtime

Register and enable manipulation

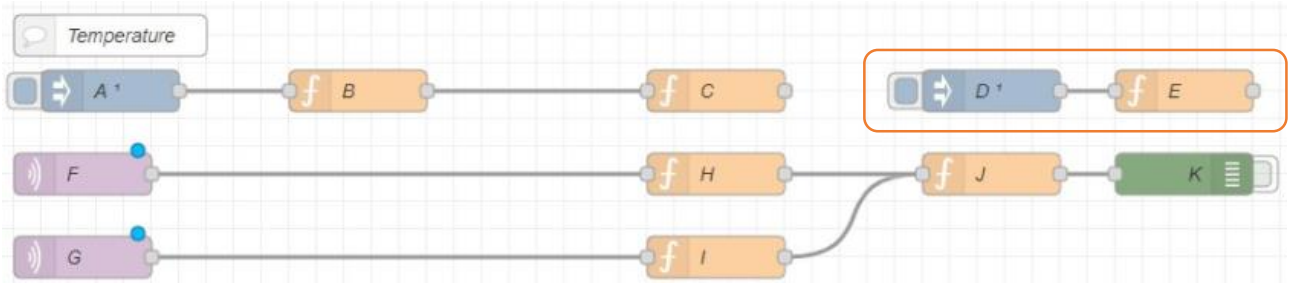
Add dimension to datasource

Apply Json

Update visualization

```
// C
try {
    console.log("INIT IATK from NR ... 1");
    var nextId = Main_GetNextChartId();

    flow.set("jsonchart", nextId);

    console.log("INIT IATK from NR ... 2");
    var chartObj = RT.VIS.GetNewChartObjectIATK2(
        nextId,
        MAIN.usercanvas.charts.length,
        "IATKChart",
        MAIN.usercanvas.canvasSize,
        MAIN.usercanvas.canvasSpacedSize,
        false);

    MAIN.usercanvas.charts.push(chartObj);
    MAIN.RR.Runtime.ToggleObjManipulation(chartObj.chartGoName);

    //chartObj.rtds.AddDimension("distance", 0, 300);
    //chartObj.rtds.SetDataStrVal("distance", 250);

    chartObj.rtds.AddDimension("temperature", 0, 15);
    chartObj.rtds.SetDataStrVal("temperature", 7);
    chartObj.abstractVisualisation.UpdateVisualisation(chartObj.PropertyType.Y);

    //apply json here
    var json = JSON.stringify(msg.payload);
    chartObj.vis.theVisualizationObject.creationConfiguration.DeserializeJson(json);

    //update from json
    importNamespace("IATK").Replicator.SyncVis(chartObj.vis);
    chart.fctUpdateChart();
    chart.vis.updateProperties();

    msg.payload = nextId;
} catch(err) {
    var strmsg = "trying to run client side code ... => " + err;
    msg.payload = strmsg;
    node.warn("C => " + strmsg);
}
return msg;
```

Smart Fridge




```
// E
var chartId = flow.get("jsonchart");
try {
    var chart = MAIN.usercanvas.charts[chartId];

    //move in front of the user
    var hT = MAIN.camHookGo.transform.position;
    var hR = MAIN.camHookGo.transform.rotation;
    var T = [hT.x, hT.y, hT.z];
    var R = [hR.x, hR.y, hR.z, hR.w];
    RT.Unity.SetPose(chart.chartGo, T, R, null);

    //ad a boxcollider to allow manipulation
    var bc = MAIN.RR.Runtime.AddBoxCollider(chart.chartGo);
    bc.size = new MAIN.UE.Vector3(0.3, 0.3, 0.3);
} catch(err) {
    node.warn("CNR E ERROR (maybe trying to run client code?) => " + err);
}
return msg;
```

Get chart ID

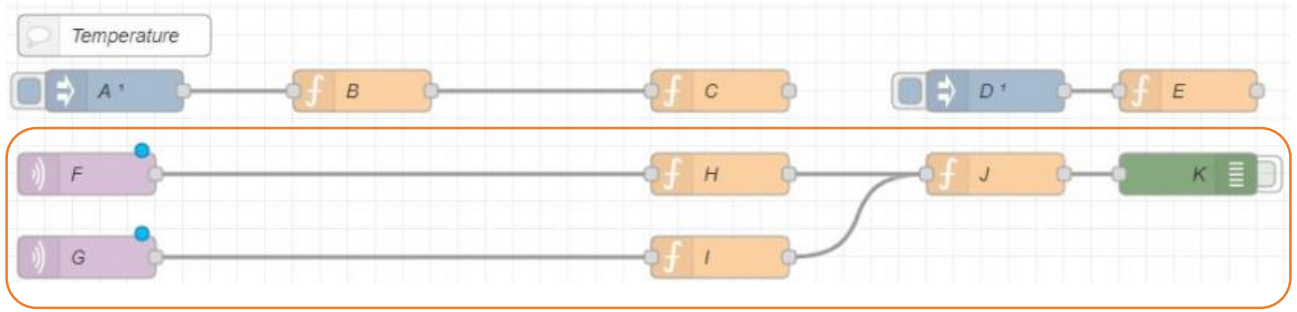
Get chart object

Get HUD Hook GameObject ➔
eg. move 1m in front of the camera

Apply pose to chart

Add box-collider for interaction

Smart Fridge



Parse JSON, pass on js-object

```
// H
var data = JSON.parse(msg.payload);
msg.payload = data;
return msg;
```

```
// I
var data = JSON.parse(msg.payload);
msg.payload = data.StatusSNS;
return msg;
```

Unwrap data

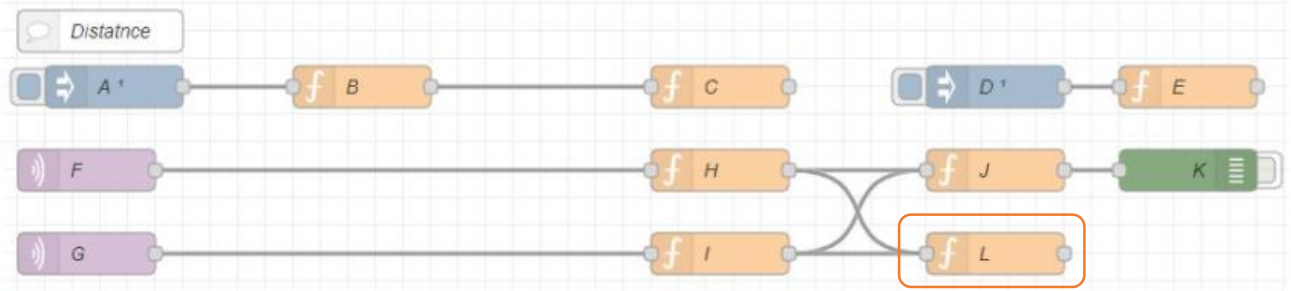
```
// J
//example payload
//"{"Time":"2021-08-17T07:31:36","BME280":{"Temperature":26.1,"Humidity":47.2,"Dew
var data = msg.payload;
var temperature = data.BME280.Temperature;
var humidity = data.BME280.Humidity;
var distance = data.SR04.Distance;
var chartId = flow.get("jsonchart");

try {
    var chart = MAIN.usercanvas.charts[chartId];
    //chart.rtds.SetDataStrVal("distance", distance);
    chart.rtds.SetDataStrVal("temperature", temperature);
    chart.fctUpdateChart();
} catch(err) {
    node.warn("CNR J ERROR (maybe trying to ru
}
return msg;
```

Extract values

Add Data to real-time datasource

Update chart

Smart Fridge



- Distance based on temperature data-flow
- Accessing distance measurement
- Feeding the distance Graph
- Turning on and off the alarm
- Playing back the alarm sound

Read distance

Evaluate distance

```
// L
var data = msg.payload;
var distance = data.SR04.Distance;
var falarm_old = flow.get("fridgealarm");
try {
    if(distance > 35) {
        PlayNotification();
        if(falarm_old == false) {
            flow.set("fridgealarm", true);
            msg.payload = true;
            RT.Unity.SpawnAlert(45, "Alarm!</br> Fridde door open!");
            return msg.payload;
        }
    } else {
        if(falarm_old == true) {
            flow.set("fridgealarm", false);
            msg.payload = false;
            var PUI = importNamespace("PowerUI");
            var wu = PUI.WorldUI.Find("WU_ALERT");
            if(wu != null) {
                wu.Expire();
            }
            return msg.payload;
        }
    }
} catch(err) {
    node.warn("CNR J ERROR (maybe trying to run client code?) => " + err);
}
return null;
```

turn on alarm

turn off alarm

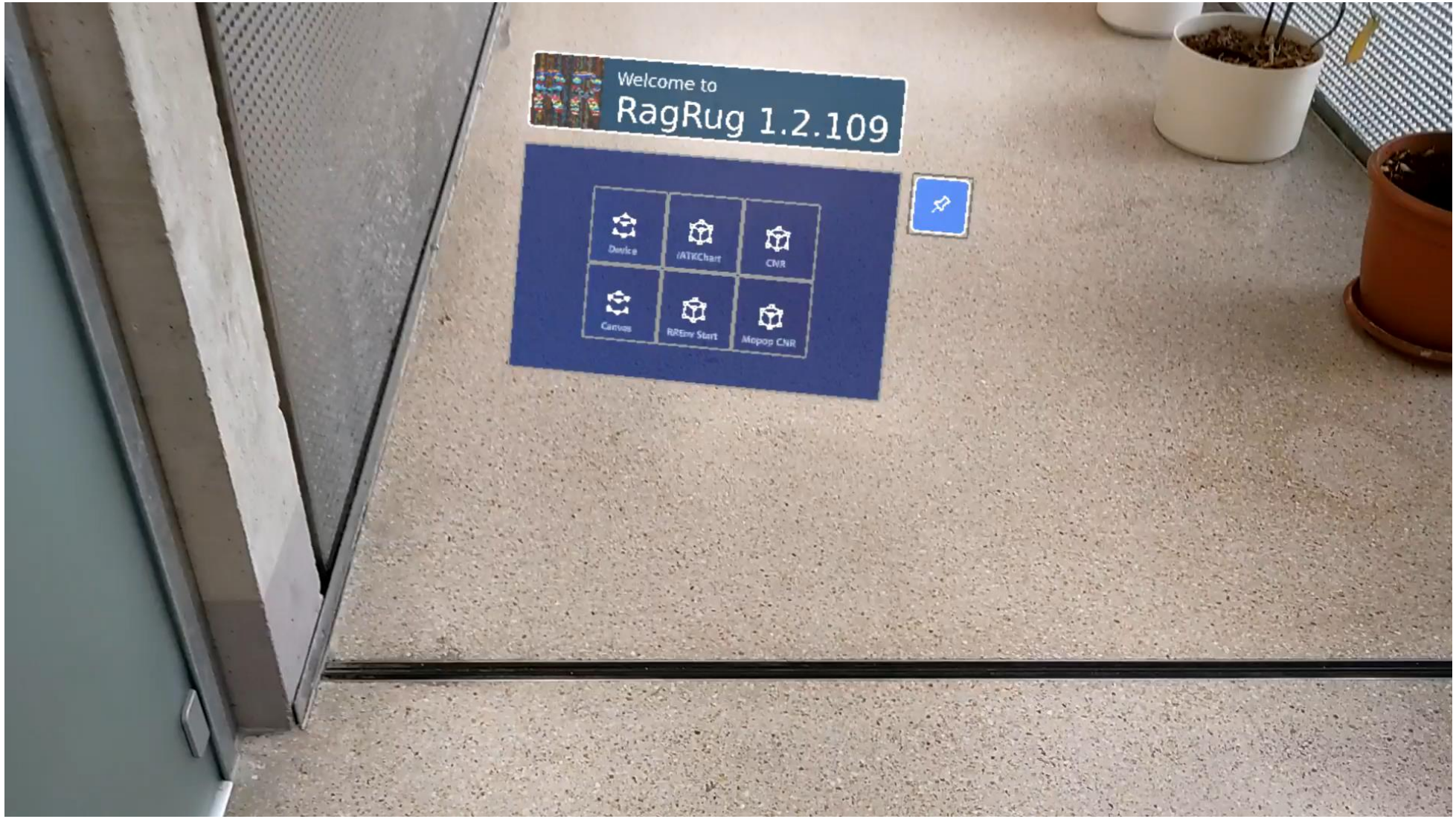
Add MRTK button for loading

Call init method To download an instatiate flow

```
// clientcode likely added insid main.js
var mopopCnrButton = RT.MRTK.SpawnButton("UTTN-loadflow", "Load flow", "Load flow", false,
  function () { CNR_InitFromId("923bd7d5.f34fa8"); });
MAIN.buttons.push(mopopCnrButton);
RT.Unity.SetParent(mopopCnrButton.go, MAIN.palmHandMenuGo);
RT.Unity.SetLocalPose(mopopCnrButton.go, [-0.04, 0.1, 0], null, null);
```

Place button in front

Smart Fridge in Action



Setting up a Location

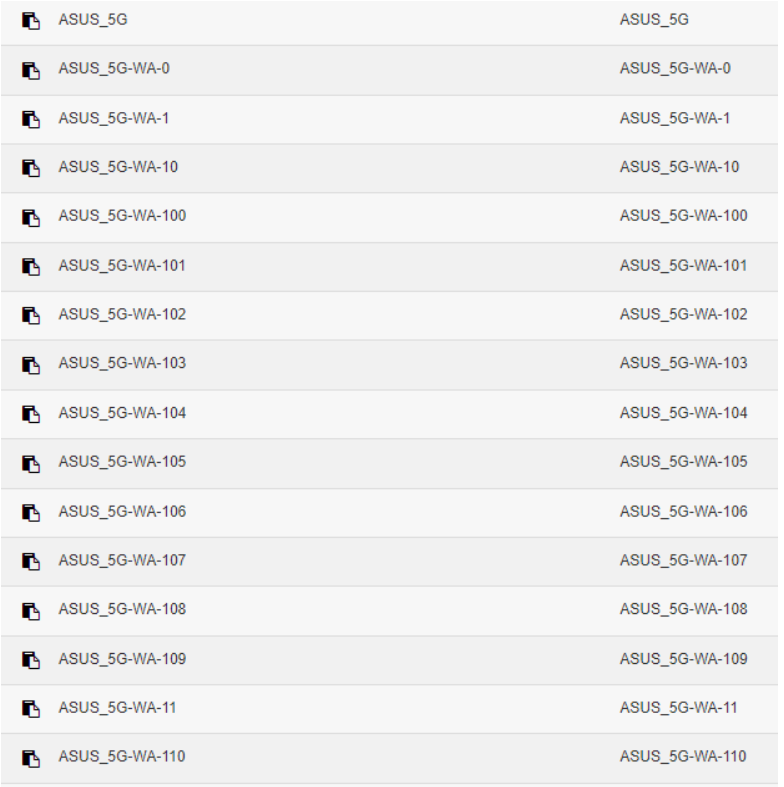
Tutorial

Tutorial overview

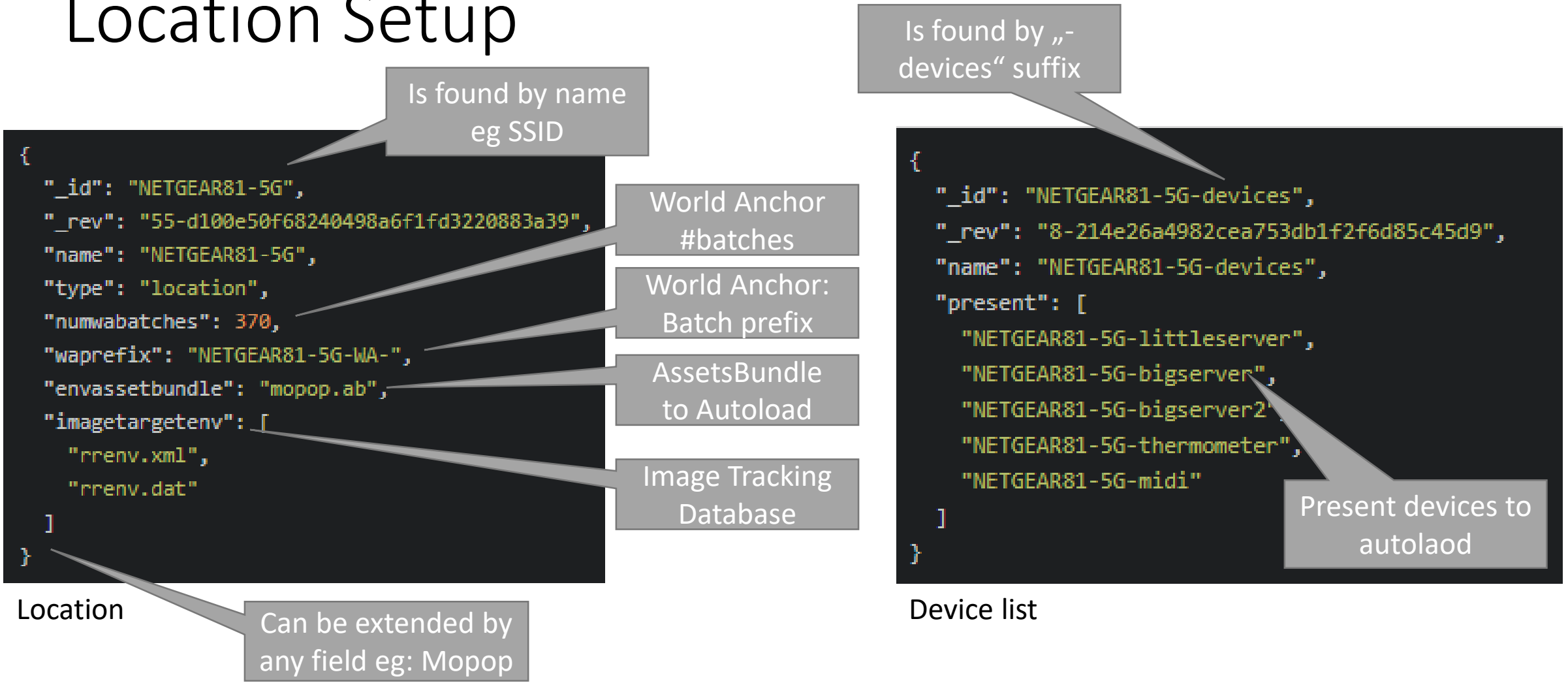
Tutorial	Visualization	Data Flow	Location	Tracking
IATK	X			
Smart Fridge	X	X		
Location			X	
Basement	X	X	X	
Target Tracking			X	X
ControllAR	X	X	X	X

Location Setup

- Location is stored in CouchDB
- Holds information about
  - Devices
  - World anchors
  - AssetBundles
  - And other location-related information (eg mopop)
- We use the convention {locationname} for the location and {locationname}-devices for the device list.
- Tways to add a location:
  - Once the ARCLient is within a new location, but it is not found, the location will be automatically added by the client with default values. Those can be changed later.
  - We can also add the location defintion manually



Location Setup




```
{
  "_id": "NETGEAR81-5G",
  "_rev": "55-d100e50f68240498a6f1fd3220883a39",
  "name": "NETGEAR81-5G",
  "type": "location",
  "numwabatches": 370,
  "waprefix": "NETGEAR81-5G-WA-",
  "envassetbundle": "mopop.ab",
  "imagetargetenv": [
    "rrenv.xml",
    "rrenv.dat"
  ]
}
```

Location

```
{
  "_id": "NETGEAR81-5G-devices",
  "_rev": "8-214e26a4982cea753db1f2f6d85c45d9",
  "name": "NETGEAR81-5G-devices",
  "present": [
    "NETGEAR81-5G-littleserver",
    "NETGEAR81-5G-bigserver",
    "NETGEAR81-5G-bigserver2",
    "NETGEAR81-5G-thermometer",
    "NETGEAR81-5G-midi"
  ]
}
```

Device list

Location Setup (Referents/Devices)

- Devices consist of parts
- Each part will spawn a leader line to its location in combination with a Label (label.html, label.js)
- Each part/label is running ist own DOM on ist one context
- We can piggy-back from there
  - eg loading cnr, adding custom functionality, …
- When registered, the device will be spawned at the registered transform
- Support for additional external url, which will be loaded on ist own DOM
- Enable tracking, either 3D oder 2D via Vuforia

N-Parts of a device

Local position and additional fields (modelfree)

Index for tracking target

Ui to be loaded on device, aligned with the tracking origin

World orientation (RTS)


```
{
  "_id": "NETGEAR81-5G-midi",
  "_rev": "8-3b2a300f43fc5e86b9c14f3358a490e3",
  "Xmodelname": "midi",
  "devicename": "midi",
  "registered": true,
  "assetname": "midi.ab",
  "commonname": "midi",
  "locationkey": "NETGEAR81-5G",
  "locationpath": "inffeld16/2nd/id2068",
  "scale": 1,
  "id": "NETGEAR81-5G-midi",
  "parts": [{
      "name": "DemoPart",
      "position": [
        0.0593402295,
        0.00209352895,
        0.0061
      ],
      "parent": "root",
      "active": true
    }],
  "envtrackedname":"machine_mikro_2",
  "ondevicecontrols": {
    "controlurl": "http://10.0.0.2:9999/midi.html",
    "width": 1,
    "height": 1
  },
  "transform": {
    "position": [
      0.2739790201187134,
      -0.34729713201522827,
      0.4392329454421997
    ],
    "rotation": [
      0.19814874231815338,
      0.742375373840332,
      0.5748777985572815,
      -0.2813031077384949
    ],
    "scale": [
      1.0000001192092896,
      0.9999998807907104,
      1.0000001192092896
    ]
  }
}
```


Device

Basement

Tutorial

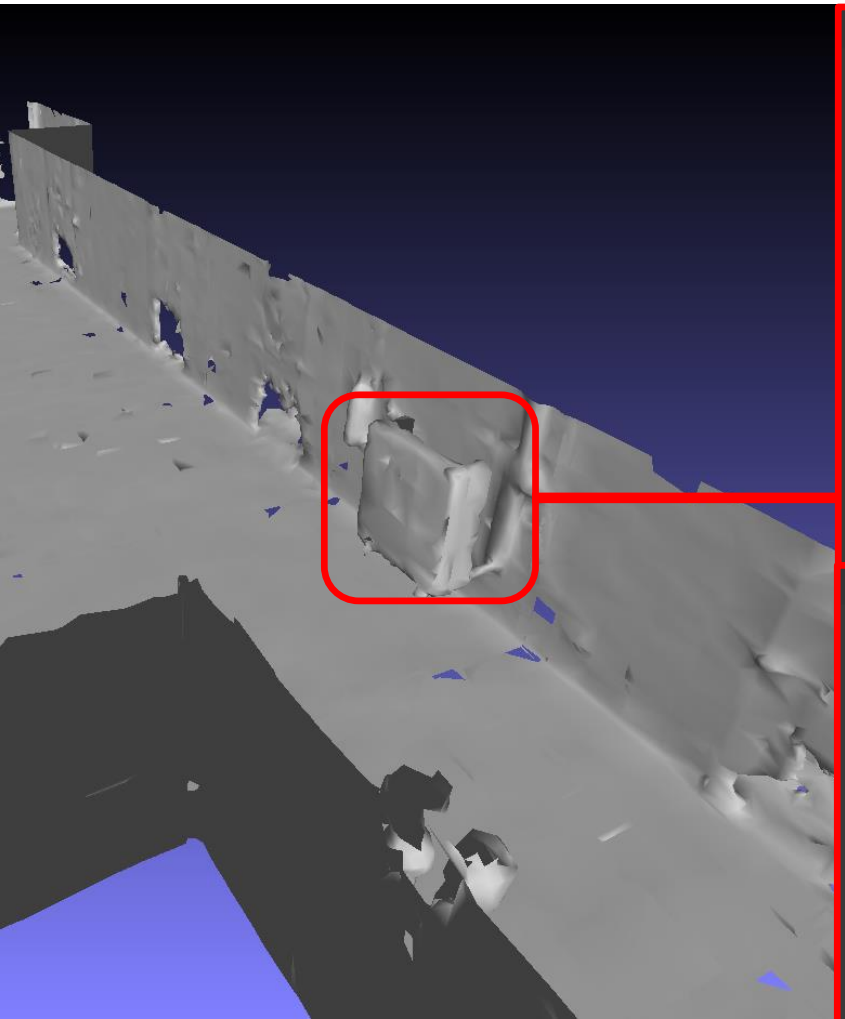
Tutorial overview

Tutorial	Visualization	Data Flow	Location	Tracking
IATK	X			
Smart Fridge	X	X		
Location			X	
Basement	X	X	X	
Target Tracking			X	X
ControllAR	X	X	X	X

Basement location

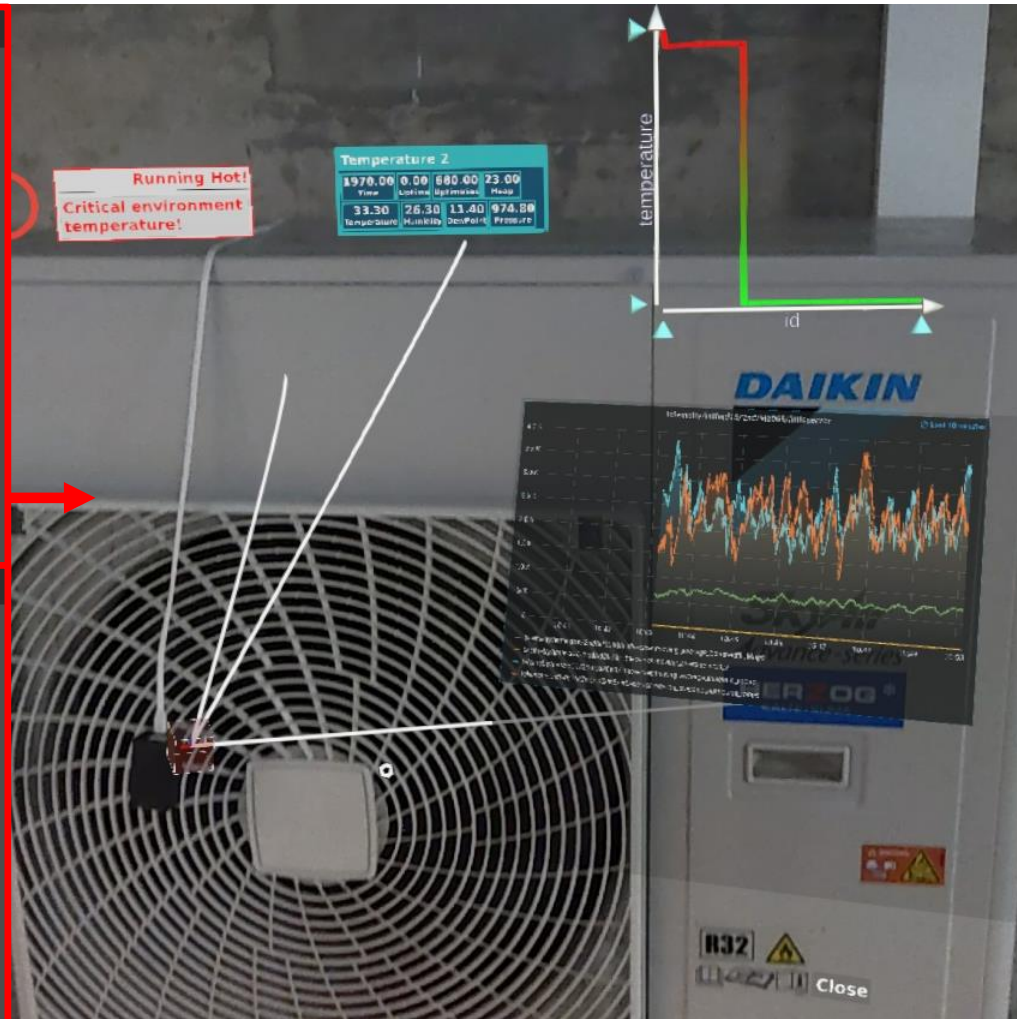
- Device definition
- Device registration within the world
  - manual within the app
  - By detection (marker, 3d model, …)
- Sensor data reading
  - Temperature, Humidity, AirPreasure
- Sensor data visualization
- Proxemics to show close by sensors in detail

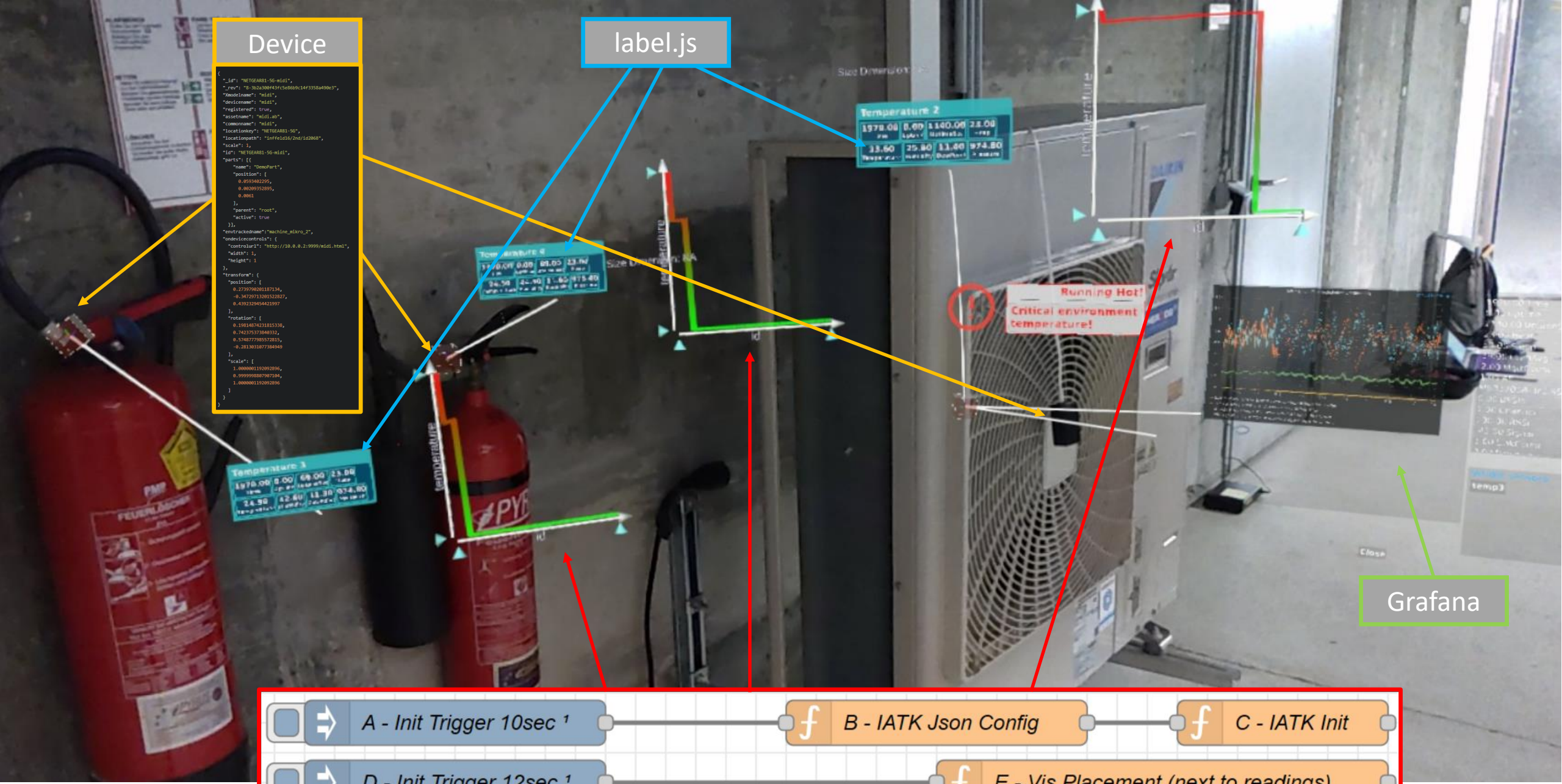
Basement Location




```
{
  "_id": "MBR1200B-3c1",
  "_rev": "3-0551c90c6eae914497fb56962c623529",
  "name": "MBR1200B-3c1",
  "type": "location",
  "numwabatches": 0,
  "waprefix": "MBR1200B-3c1-WA-",
  "Xenvassetbundle": "mopop.ab",
  "imagetargetenv": [
    "rrenv.xml",
    "rrenv.dat"
  ]
}
```

```
{
  "_id": "MBR1200B-3c1-devices",
  "_rev": "1-6397733d1ef4216ed70a154b780284aa",
  "name": "MBR1200B-3c1-devices",
  "present": [
    "MBR1200B-3c1-temp1",
    "MBR1200B-3c1-temp2",
    "MBR1200B-3c1-temp3",
    "MBR1200B-3c1-temp4"
  ]
}
```


Device
label.js
Grafana
Temperature 2
Temperature 3
Running Hot!
Critical environment temperature!

A - Init Trigger 10sec ¹
B - IATK Json Config
C - IATK Init
D - Init Trigger 12sec ¹
E - Vis Placement (next to readings)
F - telemetry/inffeld16/basement/hallway/#
G - Auto Topic Filter
J - Vis Update

Add Vuforia Tracking for Location

Tutorial

Tutorial overview

Tutorial	Visualization	Data Flow	Location	Tracking
IATK	X			
Smart Fridge	X	X		
Location			X	
Basement	X	X	X	
Target Tracking			X	X
ControllAR	X	X	X	X

Vuforia Tracking for Location

- Depending on the application, we can add vuforia tracking database as a known key within the device / or location definition in couchdb or createan arbitrary key, which can later be used
- Adding imagetargetenv to the location, to be loaded at startup when unsing the RR-Runtime
- The order is important, the xml goes first.Within main.js it will end up here

```
"imagetargetenv": [
    "rrenv.xml",
    "rrenv.dat"
  ]
```

Addin to location

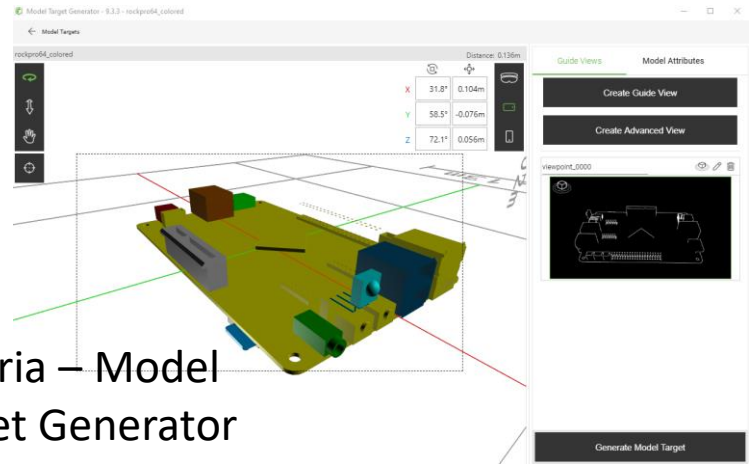
AND / OR

```
"trackingdb": [
    "rockpro64_clean_centered_scaled.xml",
    "rockpro64_clean_centered_scaled.dat"
  ]
```

Addin to device

Eg: 3d tracking database created with ModelTargetGenerator

Target Name	Type	Rating	Status	Date Modified
28543458862	Single Image	★★★★★	Active	Mar 02, 2021 20:33
25200627317	Single Image	★★★★★	Active	Mar 02, 2021 20:33
24794519636	Single Image	★★★★★	Active	Mar 02, 2021 20:33



Vuforia – Model Target Generator

Vuforia Tracking for Location

```
if (data.hasOwnProperty("imagetargetenv")) {
    var envTargets = data.imagetargetenv;
    MAIN.location.envtargets = {
        "xml": envTargets[0],
        "dat": envTargets[1]
    };
}
```

Check and load if location has tracking enabled

Download tracking db

```
function MAIN_StartRREnvTracking() {

    var reqHeaders = ["content-type", "application/x-www-form-urlencoded"];
    var url = MAIN.WEBAPI.apiBase + "/getfile2?name=";

    var urlA = url + MAIN.location.envtargets.xml;
    RT.Web.DownloadFile("GET", urlA, reqHeaders, MAIN.location.envtargets.xml, false, function () {
        console.log("MAIN_StartRREnvTracking downloaded => " + MAIN.location.envtargets.xml);
    });

    var urlB = url + MAIN.location.envtargets.dat;
    RT.Web.DownloadFile("GET", urlB, reqHeaders, MAIN.location.envtargets.dat, false, function () {
        console.log("MAIN_StartRREnvTracking downloaded => " + MAIN.location.envtargets.dat);
    });

    // Start tracking
    importNamespace("RR").Runtime.StartImgTracking(
        MAIN_RREnvOnTracked, MAIN_RREnvOnTrackingLost, MAIN.location.envtargets.xml);

    MAIN_RREnvStartPosUpdater();
}
```

Start tracking

On tracked callback

On tracking lost callback

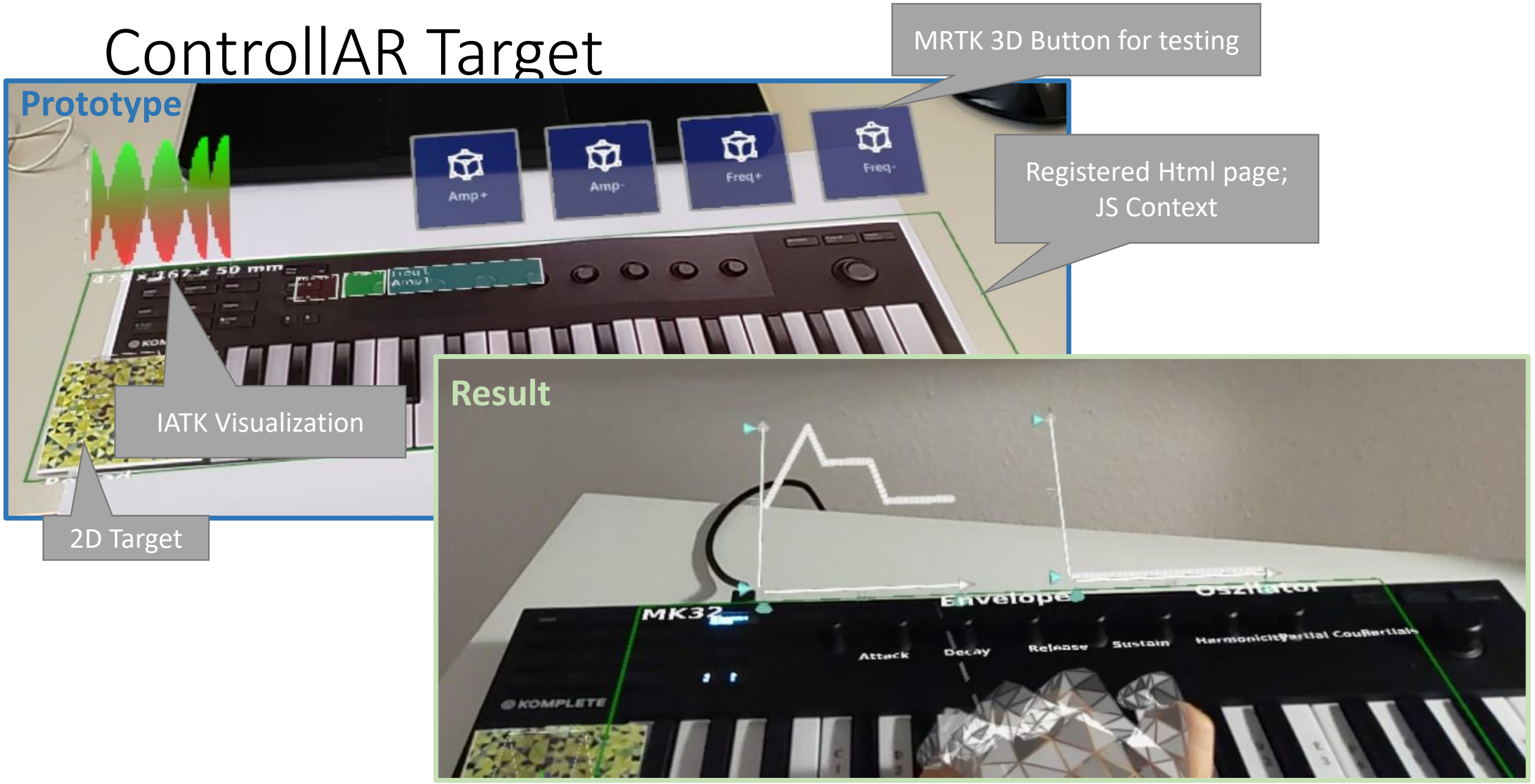
Tracking database

MIDI Device (ControllAR)

Turial overview

Tutorial	Visualization	Data Flow	Location	Tracking
IATK	X			
Smart Fridge	X	X		
Location			X	
Basement	X	X	X	
Target Tracking			X	X
ControllAR	X	X	X	X

ControllAR Target



Mapping the Layout

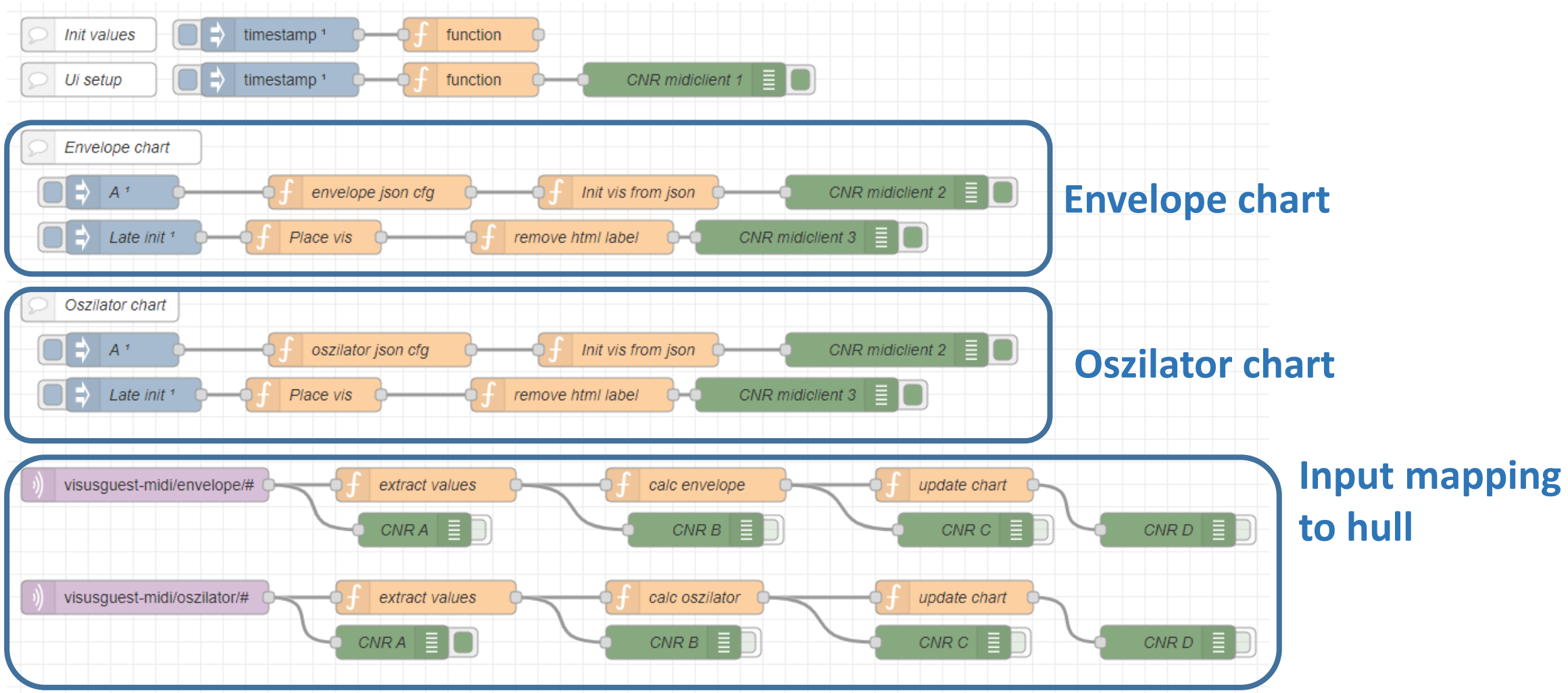
- Calculating the rendering size to fit real-world size
- By default we create HTML Dom with an resolution 1920px x 1080px where the default ppw = 0.35 this is set to render nicely within the hololens.

```
The board is 475 x 167 x 50 mm

p[px] / ppw = m[m]
1920px / 0,35m = 5485 // default 1920px @ 35cm width
X / 0,475m = 5485 => 0,475 * 5485 = 2605px
Y / 0,167 = 5485 => 0,167 * 5485 = 915px

By keeping the resolution we need to render at 2605px to get a
physical size of 47,5cm and height of 915px to result in 16,7cm.
```

Device extension

```
"ondevicecontrols": {
    "controlurl": "http://10.0.0.2:9999/midisim.html",
    "width": 1,
    "height": 1,
    "pixelw": 2605,
    "pixelh": 915
  }
```

Client NodeRed Flow

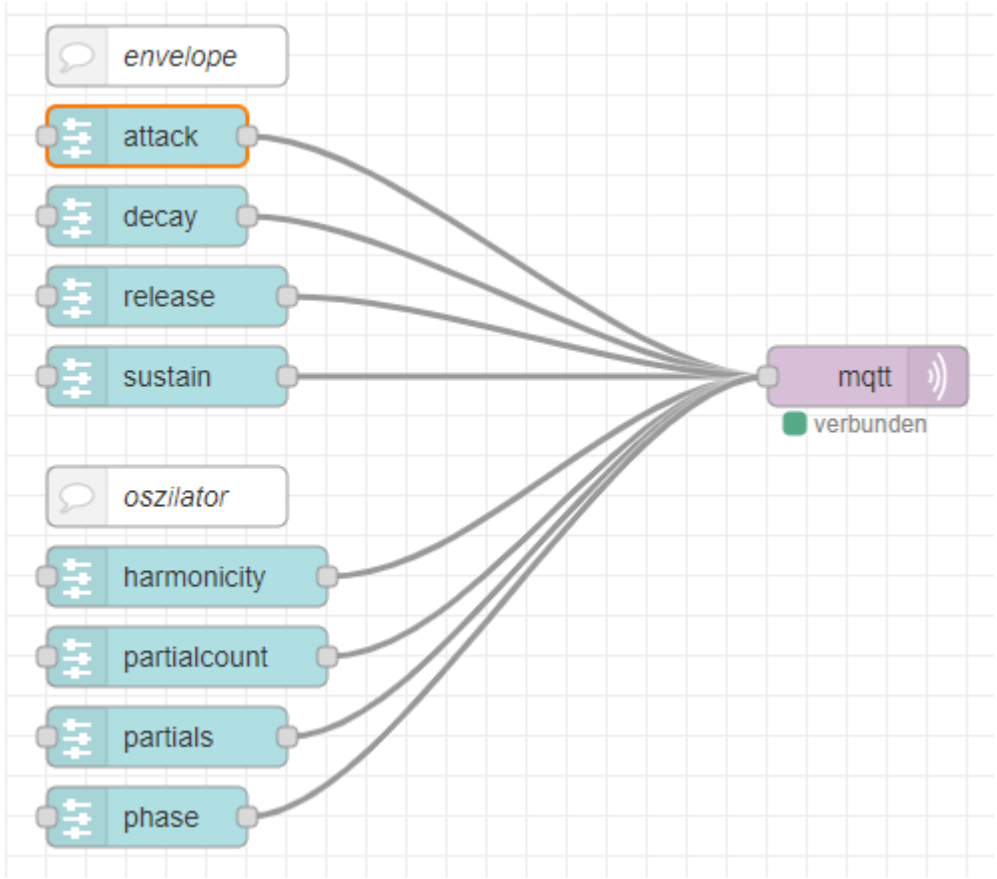


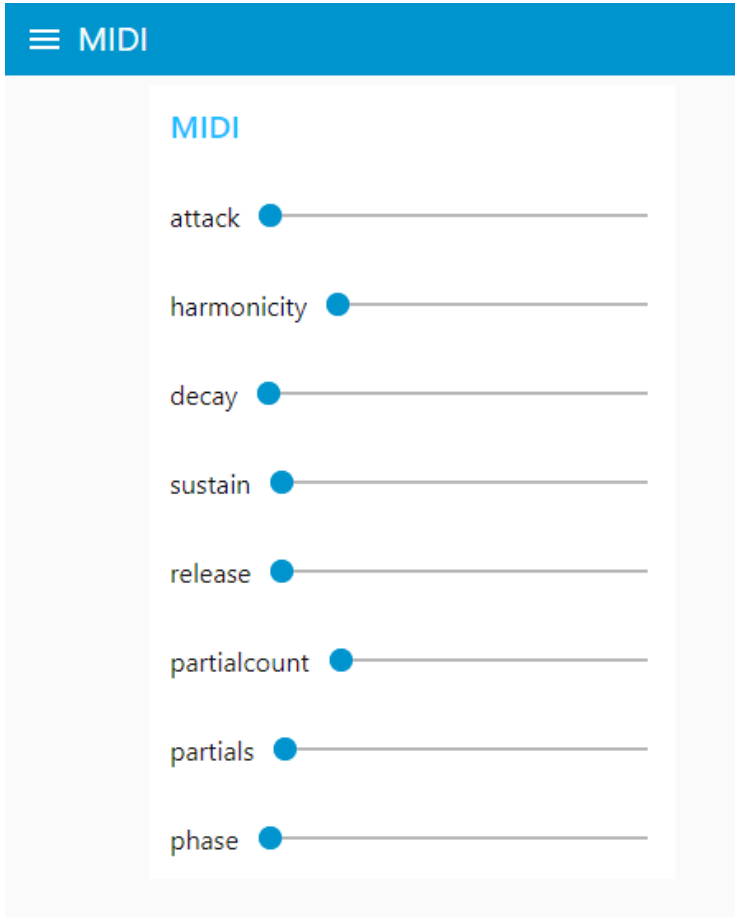
Envelope chart

Oszilator chart

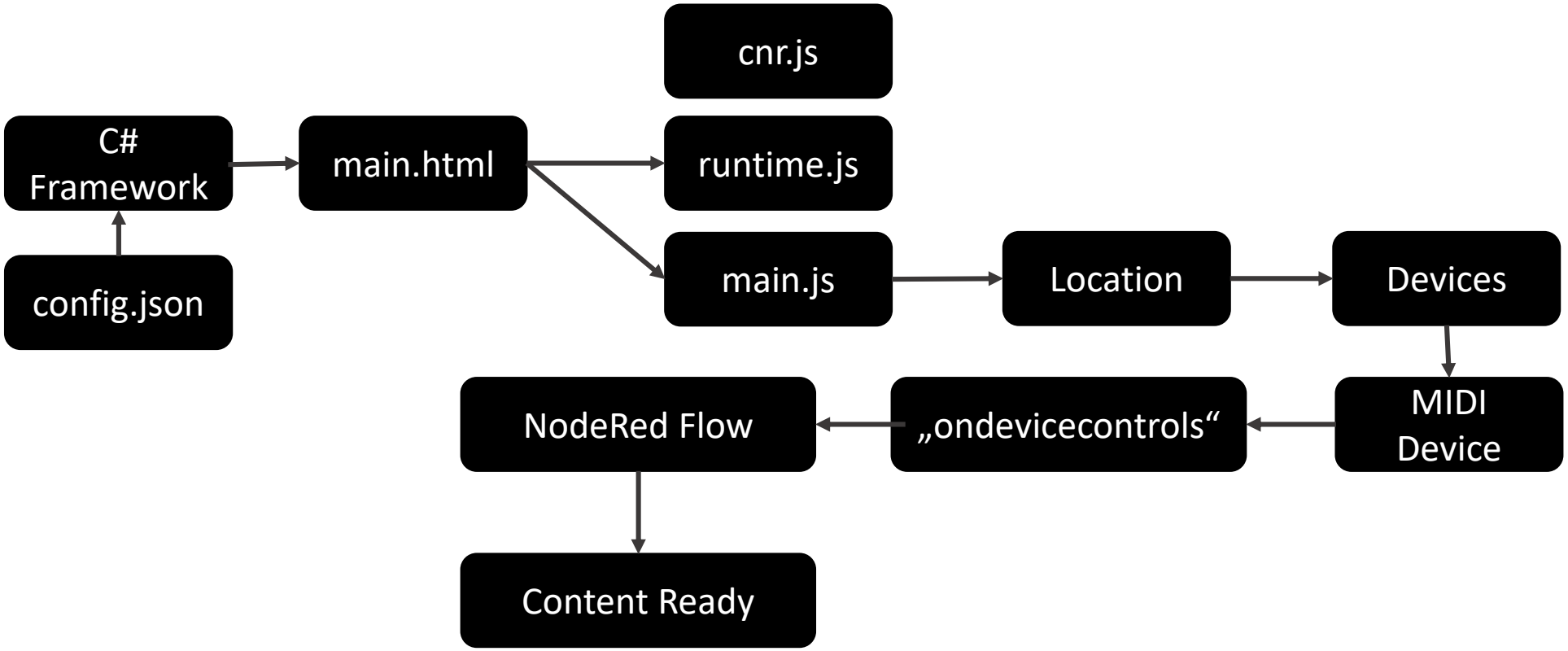
Input mapping to hull

Debug / Test interface (ServerSide)





How is the loading done?




KOMPLETE

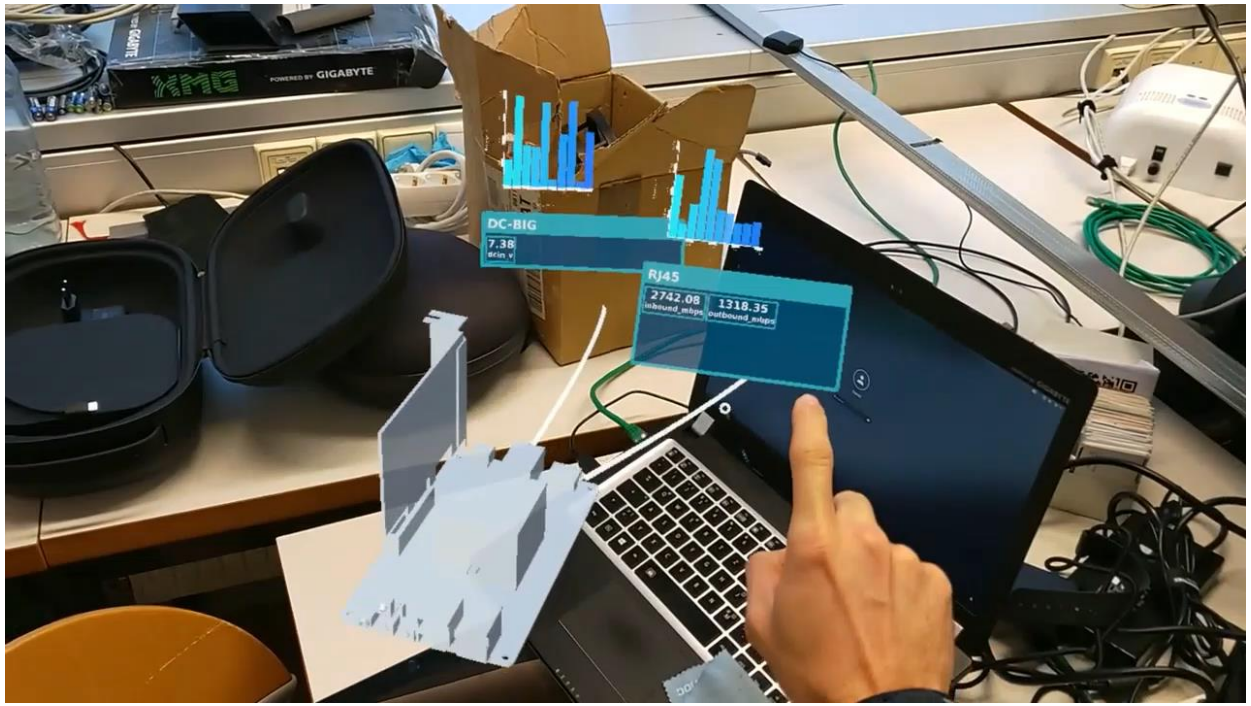
DI Philipp Fleck
philipp.fleck@icg.tugraz.at
Graz University of Technology
Institute of Computer Graphics and Vision

Thank you for your Attention!

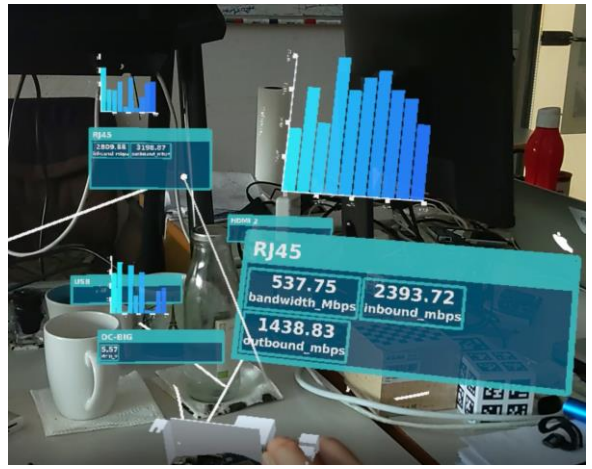
Interaction: Sticky finger

- Draging (Tap-and-Hold to drag) in AR can be very hard, especiality through space or along different surfaces
  - Eg Minority Report Style (mostly uses one surface)

- Tap-to-Pickup / Tap-To-Drop
  - Hololens 2 Handtracking with



Adding data to Vis-Template instances



- Parts with abilities → producers / Sources
  - Clicking on a value (sticky finger)
- Dimension of a visualization → consumer / sink
  - Adding to visualization via „Add Dimension“
- Linking via MQTT subscription
- Every Visualization holds its Dimension-Data (data table)
- By adding a new dimension, we add a new column and subscribe to the source.
- New incoming data is added Fifo/Circle Buffer style (+ ability to shift or scroll indices)