**Documentación del proyecto**

1. **Estructura del proyecto**

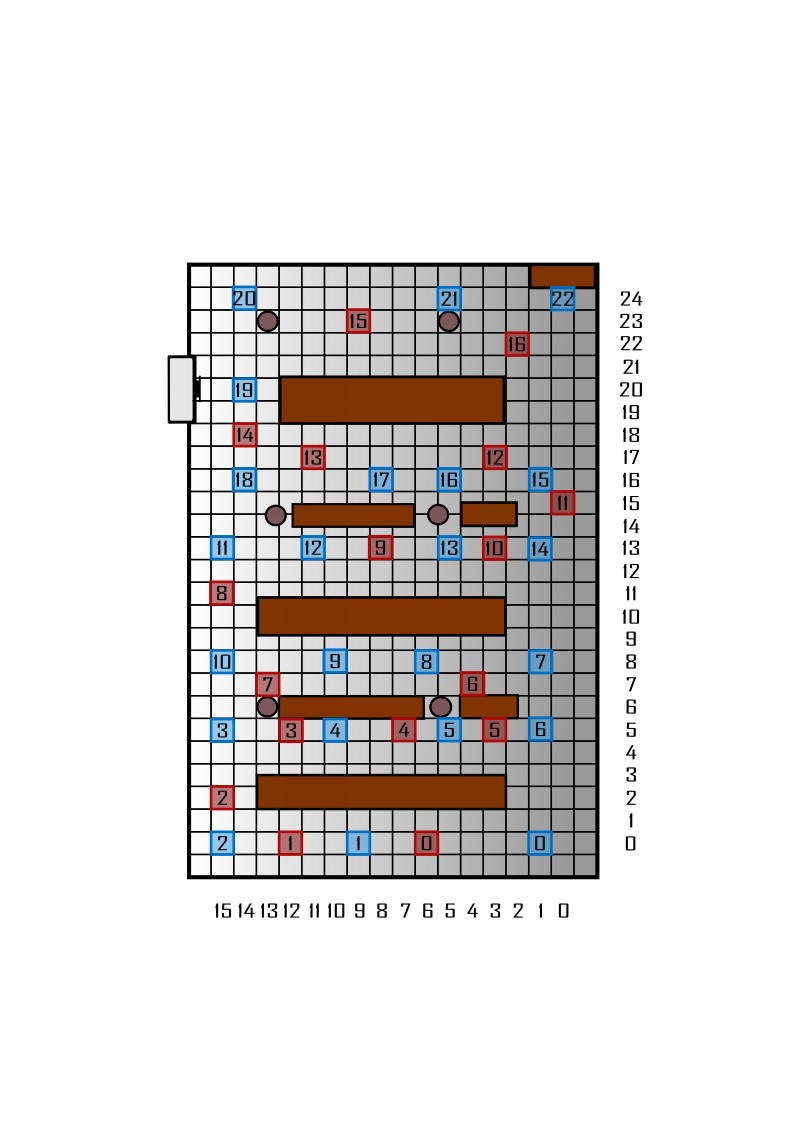
Texto

Descripción generada automáticamente con confianza mediaLa estructura entera del proyecto consiste en 5 directorios principales

* + **Data**: Directorio en la que se encuentran tanto los datos obtenidos usando la aplicación **get\_sensordata**, como los datos preprocesados
  + **Info**: Directorio con la siguiente información relacionada con el proyecto:
    - Fichero con documentación del proyecto
    - Fichero con las coordenadas usadas en cada punto de referencia de Train
    - Fichero con las coordenadas usadas en cada punto de referencia de Test
    - Mapa del plano en la recogida de datos, indicando la posición de cada punto de referencia, tanto en entrenamiento como en test
  + **Outputs:** Como indica su nombre, directorio de salidas:
    - Imágenes
    - Checkpoints de modelos (importante)
    - Tablas con resultados y métricas
    - Gifs
  + **src:** Directorio que incluye scripts de python con métodos y clases fundamentales en todo el proceso de desarrollo del proyecto. Con funciones y clases de preprocesado, herramientas de visualización, carga de datos y **CONSTANTES** del proyecto.

Incluye también, los scripts esenciales para ejecutar en procesos el proyecto, los cuales son ejecutados desde ***obtainDataINITandPositioning.py***

1. Datos del proyecto, preprocesado y estructuración
   1. Datos de Train y Test



Los datos se han recogido manualmente, utilizando la aplicación **get\_sensordata**, en el edificio del INIT. La imagen de arriba muestra un mapa de la planta y de la localización de los distintos puntos de referencia de recogida de datos. El color azul representa los puntos de Train, mientras que el color rojo representa los puntos de Test.

Se genera un fichero por cada punto de referencia y, una vez recogidos los datos con algún dispositivo Android (móvil, Tablet,…), se traspasan manualmente al siguiente directorio (en train y en test):

Patrón de fondo

Descripción generada automáticamente

data  
|----train  
| |---- initial\_rp\_data (TRAIN DATA)  
|

|----test

| |---- initial\_rp\_data (TEST DATA)

Si consideramos que cada baldosa, representada en las rejillas grises de la imagen, mide 60 centímetros, entonces realizamos una transformación de coordenadas ficticias (baldosas) a coordenadas longitud, latitud (metros). La tabla la tenemos en la siguiente página

**Tabla de puntos de referencia con sus coordenadas**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Label | Longitud | Latitud |
| TRAIN | 0 | 0.6 | 0 |
| 1 | 5.4 | 0 |
| 2 | 9 | 0 |
| 3 | 9 | 3 |
| 4 | 6 | 3 |
| 5 | 3 | 3 |
| 6 | 0.6 | 3 |
| 7 | 0.6 | 4.8 |
| 8 | 3.6 | 4.8 |
| 9 | 6 | 4.8 |
| 10 | 9 | 4.8 |
| 11 | 9 | 7.8 |
| 12 | 0.6 | 7.8 |
| 13 | 3 | 7.8 |
| 14 | 0.6 | 7.8 |
| 15 | 0.6 | 9.6 |
| 16 | 3 | 9.6 |
| 17 | 4.8 | 9.6 |
| 18 | 8.4 | 9.6 |
| 19 | 8.4 | 12 |
| 20 | 8.4 | 14.4 |
| 21 | 3 | 14.4 |
| 22 | 0 | 14.4 |
| TEST | 0 | 3.6 | 0 |
| 1 | 7.2 | 0 |
| 2 | 9 | 1.2 |
| 3 | 7.2 | 3 |
| 4 | 4.2 | 3 |
| 5 | 1.8 | 3 |
| 6 | 2.4 | 4.2 |
| 7 | 7.8 | 4.2 |
| 8 | 9 | 6.6 |
| 9 | 4.8 | 7.8 |
| 10 | 1.8 | 7.8 |
| 11 | 0 | 9 |
| 12 | 1.8 | 10.2 |
| 13 | 6.6 | 10.2 |
| 14 | 8.4 | 10.8 |
| 15 | 5.4 | 13.8 |
| 16 | 1.2 | 13.2 |

En el directorio principal tenemos dos scripts de Python: ***process\_train.py*** y ***process\_test.py***. La ejecución de cada uno de estos scripts nos va a realizar todo el preprocesado necesario para obtener el *radiomap* preparado para el entrenamiento de los modelos de posicionamiento.

La ejecución de estos scripts nos proporcionará el siguiente directorio de salida:

(nuevos ficheros y directorios subrallados)

data

|----train

| |---- initial\_rp\_data

| |---- checkpoint\_groundtruth (process\_train.py execution)

| |---- raw\_radiomap (process\_train.py execution)

| | |---- raw\_radiomap.csv (process\_train.py execution)

| |---- processed\_radiomap (process\_train.py execution)

| | |----- processed\_radiomap.csv (process\_train.py execution)

|

|---- test

| |---- initial\_rp\_data

| |---- checkpoint\_groundtruth (process\_test.py execution)

| |---- raw\_radiomap (process\_test.py execution)

| | |---- raw\_radiomap.csv (process\_test.py execution)

| |---- processed\_radiomap (process\_test.py execution)

| | |----- processed\_radiomap.csv (process\_test.py execution)

Donde ***raw\_radiomap.csv*** corresponde con el radiomap sin escalado de los datos, por lo que se presenta en unidades (decibelios), mientras que ***processed\_radiomap.csv*** presenta los niveles de RSS escalados entre 0 y 1.

Los scripts de preprocesado, dependen directamente de los métodos implementados en los scripts **preprocess.py** y **constants.py** que se encuentran en el directorio src

root

|

|----data

|---- src

| |----preprocess.py

| |----constants.py

Dentro de las constantes, en el script **constants.py**, es importante definir las siguientes de manera correcta, y adecuado para el sistema de directorios en el trabajo, para que el preprocesado se realice de manera correcta:

**constants.data.train.INITIAL\_DATA:** dirección de los datos iniciales de entrenamiento

**constants.data.train.CHECKPOINT\_DATA\_PATH:** dirección de los checkpoints de los datos de entrenamiento

**constants.data.train.RAW\_OUT\_PATH:** dirección del radiomap en bruto de entrenamiento

**constants.data.train.PROC\_OUT\_PATH:** dirección del radiomap escalado de entrenamiento

**constants.data.train.INITIAL\_DATA:** dirección de los datos iniciales de test

**constants.data.train.CHECKPOINT\_DATA\_PATH:** dirección de los checkpoints de los datos de test

**constants.data.train.RAW\_OUT\_PATH:** dirección del radiomap en bruto de test

**constants.data.train.PROC\_OUT\_PATH:** dirección del radiomap escalado de test

**constants.aps:** lista con los puntos de acceso WiFi a considerar para la obtención del fingerprint

**constants.labels\_dictionary\_meters:** diccionario que transforma de label a coordenadas en entrenamiento

**constants.labels\_dictionary\_meters\_test:** diccionario que transforma de label a coordenadas en test

**constants.labels\_train:** lista con las labels (puntos de acceso) a considerar en entrenamiento

**constants.labels\_test:** lista con las labels (puntos de acceso) a considerar en test

**constants.T\_MAX\_SAMPLING:** tiempo máximo (en segundos) de muestreo en cada label de entrenamiento

**constants.T\_MAX\_SAMPLING\_TEST:** tiempo máximo (en segundos) de muestreo en cada label de test

Por último, haciendo uso de los métodos descritos en **preprocess.py**, al ejecutar los scripts **process\_train.py** y **process\_test.py**. El flujo de ejecución será el siguiente:

Diagrama

Descripción generada automáticamente

El proceso se resume en lo siguiente:

Se procesan los ficheros log de get\_sensordata, para transformarlos a un formato en el que cada fila representa el fingerprint de un segundo, como la media de todas las observaciones en ese periodo de RSS para cada AP.

Posteriormente, se cambian los NAs resultantes por el valor mínimo global – 1, esto se hace para que posteriormente, con el escalado, el 0 represente ausencia de RSS.

También se aplica un promediado móvil en ventanas de 30 segundos con un overlapping de 5 segundos, de este modo, suavizamos los valores obtenidos en el RSS.

Finalmente, se obtiene el fichero de **raw\_radiomap.csv** (sin escalado), y **processed\_radiomap.csv** (con escalado)

* 1. **Datos de particiones en Train**

También se ha planteado en el proyecto, para probar otras alternativas, ha realizar varias particiones de train y de test con los mismos datos recogidos en train, para poder estudiar los efectos en recogidas de datos de tiempos similares.

Para ello, se ha implementado el script **process\_partitions.py**. El objetivo de este script será obtener el siguiente directorio de salida aplicando los mismos pasos de preprocesado que en el caso de train, pero con puntos de referencia distintos.

El script devuelve el siguiente directorio:

data

|----train

|---- test

|---- partitions (***process\_partitions.py*** execution)

| |---- partition\_5vs18

| | |---- train

| | | |---- raw

| | | | |---- raw\_radiomap.csv

| | | |---- processed

| | | | |---- processed\_radiomap.csv

| | |---- test

| | | |---- raw

| | | | |---- raw\_radiomap.csv

| | | |---- processed

| | | | |---- processed\_radiomap.csv

| |---- partition\_10vs13

| | |---- train

| | | |---- raw

| | | | |---- raw\_radiomap.csv

| | | |---- processed

| | | | |---- processed\_radiomap.csv

| | |---- test

| | | |---- raw

| | | | |---- raw\_radiomap.csv

| | | |---- processed

| | | | |---- processed\_radiomap.csv

| |---- partition\_15vs8

| | |---- train

| | | |---- raw

| | | | |---- raw\_radiomap.csv

| | | |---- processed

| | | | |---- processed\_radiomap.csv

| | |---- test

| | | |---- raw

| | | | |---- raw\_radiomap.csv

| | | |---- processed

| | | | |---- processed\_radiomap.csv

En este paso se han planteado 3 alternativas con los datos de train, representadas en la siguiente tabla:

|  |  |  |
| --- | --- | --- |
| Partición | Conjunto | Puntos de referencia asociados |
| Partición 5 Train frente 18 Test | Train | [0, 2, 11, 14, 21] |
| Test | [1, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 15, 16, 17, 18, 19, 20, 22] |
| Partición 10 Train frente 13 Test | Train | [0, 2, 3, 5, 9, 13, 19, 22] |
| Test | [1, 4, 6, 7, 8, 10, 11, 12, 14, 15, 16, 17, 18, 20, 21] |
| Partición 15 Train frente a 8 Test | Train | [0, 2, 4, 5, 7, 9, 10, 11, 14, 15, 17, 18, 20, 21, 22] |
| Test | [1, 3, 6, 8, 12, 13, 16, 19] |

Para el correcto funcionamiento del script ***process\_partitions.py***. Hay que asegurarse que las siguientes constantes en **src/*constants.py*** estén correctamente definidas:

**constants.labels\_partition\_5vs18:** lista con puntos de referencia de train a usar en la partición 5 vs 18

**constants.labels\_partition\_10vs13:** lista con puntos de referencia de train a usar en la partición 10 vs 13

**constants.labels\_partition\_15vs8:** lista con puntos de referencia de train a usar en la partición 15 vs 8

**constants.data.partitions.PARTITION\_5VS18:** directorio de salida de la partición 5 vs 18

**constants.data.partitions.PARTITION\_10VS13:** directorio de salida de la partición 10 vs 13

**constants.data.partitions.PARTITION\_15VS8:** directorio de salida de la partición 15 vs 8