AirspaCells

Divide a regions airspace (U-space) into rectangular cells based on definable rules.

Process

1. Load a set of rules defining which ground classes and airspace types should be considered and how they impact the cells (i.e. by attaching limitations on who can enter, performance requirements, allowed altitude ranges and capacity factors to them)
2. Prepare a region that shall be segmented, this is done e.g. by supplying its name and automatically downloading the region's bounds (using OSM geocoding), then wrap these bounds into a rectilinear `Shapely` polygon
3. Gather the ground data for this region from the OSM API, identify features of interest and wrap them into categorized, rectilinear cells according to the ruleset
4. Repeat the above step for the airspace data for this region (supplied as a OpenAIR format file)
5. Dissolve neighboring and intersecting cells of the same class into combined polygons
6. Overlapping cells are checked and modified, so that only the most severe cell class (going by the defined order in the ruleset) remains for each point
7. Everything that is not covered by now is designated as a single polygon with the lowest severity class, the existing cells are left out as holes in that polygon
8. All polygons are dissected into the minimum amount of rectangles

Relevant U-space services

To perform the airspace segmentation, a number of planned U-space services would be needed. As these do not exist yet, simplified stubs are included in this tool that replace them by providing the required data from publicly available sources. The implemented services (and my understanding how they interact with each other and the airspace segmentation service) are:

- Geo-awareness (**partially implemented**): takes the input data from the services below and process them to inform other services (here the airspace segmentation) and operators
- Geospatial information (**partially implemented**): two-dimensional geo data is taken from OSM and used as a static input to identify ground features of interest
- Population density (**to be added**): is not really considered right now, would greatly enhance the ground risk classes for the identified OSM features
- Drone aeronautical information (**partially implemented**): a snapshot of the current airspace structure is taken from OpenAIP data and used as a static input
- Dynamic geofencing (**to be added**): geofences should be added as a dynamic input to update the impacted cells
- Weather information (**to be added**): wind turbulence hot spots should be added dynamically to geofence them or adapt the capacity factor and performance requirements

Usage

Initial set of cells

Run the `segmentation.py` file to perform the initial (offline) segmentation. This automatically saves the cells into a `cells.geojson` (which is an exported `GeoDataFrame`) in your current working directory.

Read cell data

To load this file back into a `GeoDataFrame` use:

```
cells = gpd.read_file("cells.geojson", driver="GeoJSON"))
```

Divide a cell

To divide a cell at the index `idx` in the `DataFrame` `cells` run:

```
cells = split_cell(cells, idx, splitter)
```

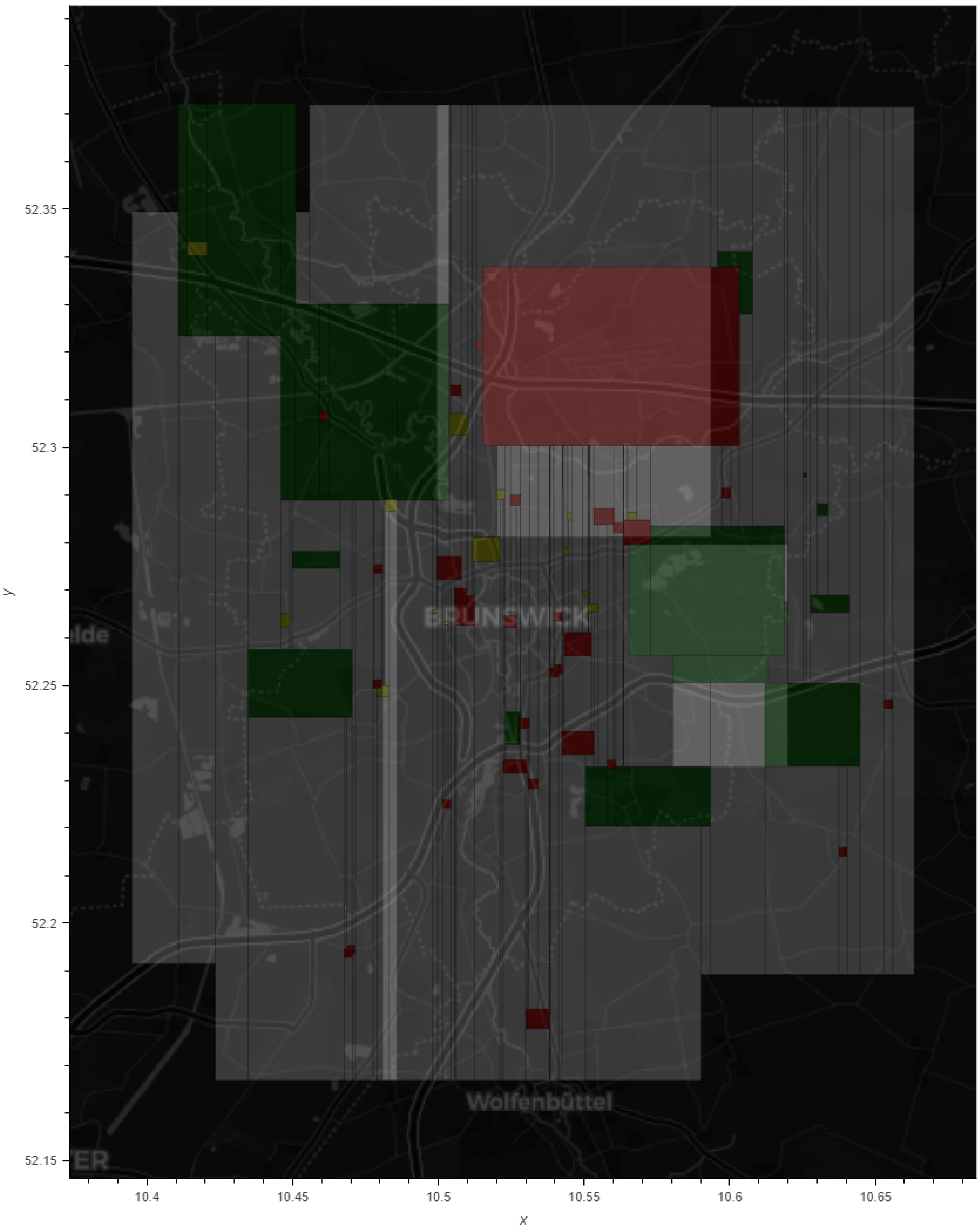
where splitter can either be `"x"`, `"y"` (both dividing the cell in half for the given direction), or any `Shapely` geometry.

Open Issues

- The dissection of the encompassing, lowest class, cell seems to create overlapping cells in a few places, so either the dissection algorithm needs to be more robust or a post-processing step could be added to clean up the rectangles

Examples

The cells below were created for the city of Braunschweig:



Generally, the DataFrames contain two columns `parent` and `children` that basically make up a linked list of how a cell was divided. If `children` is not `None`, the `parent` cell is not active anymore and has been replaced by its divided children. So for example if a cell has been subdivided using `cells = update.split(cells, 42, 'x')` and we select the parent and all its children using `cells.loc[[42] + cells.loc[42, 'children']]` we receive the following view:

	class	floor	ceiling	parent	children	geometry

	class	floor	ceiling	parent	children	geometry
42	white	0	500		[1221, 1222]	POLYGON ((9.7... 52.4..., ...))
1221	white	0	500	42		POLYGON ((9.7... 52.4..., ...))
1222	white	0	500	42		POLYGON ((9.7... 52.4..., ...))

TODO

- [x] Create initial set of cells (offline)
- [ ] Consider micro-weather in the cell creation process
- [ ] Provide methods to update cells with minimal impact if e.g. geofences are introduced
- [ ] Provide methods to dynamically split and join cells to adapt their size to traffic demand
- [ ] Check environmental protection classes (see https://wiki.openstreetmap.org/wiki/DE:Key:protect_class and https://wiki.openstreetmap.org/wiki/DE:Key:protection_title)