

Generation of magnified raster tiles using shift tiles to improve situational awareness

Tino Flenker

*Institute for the Protection of Maritime Infrastructures
German Aerospace Center
Bremerhaven, Germany
Tino.Flenker@dlr.de*

Jannis Stoppe

*Institute for the Protection of Maritime Infrastructures
German Aerospace Center
Bremerhaven, Germany
Jannis.Stoppe@dlr.de*

Abstract—Maritime infrastructures are an important part of society in many aspects (safety and security, economy, environment) and therefore require special protection. From ensuring uninterrupted global trade and economic stability to safeguarding national security and protecting coastal communities, the comprehensive situational awareness picture plays a vital role in preserving the safety and security of these vital assets.

To improve the operators' situational awareness, this work suggests a novel way to provide non-linear scaling of maps to enable legacy front-ends to provide a high resolution view of disconnected areas without requiring the user to change their viewport. So-called shift tiles are generated to scale different parts of given maps, allowing users to apply the desired scaling effect to any base map, regardless of the client being used.

Index Terms—maritime infrastructures, situational awareness, GIS, maps, tiles, shift tiles, raster tiles, magnified maps

I. INTRODUCTION

Maritime infrastructures hold significant societal importance, encompassing aspects such as the economy, the environment, and safety and security. The fact that more than 80% of global trade volume is transported by ships highlights the critical role of maritime infrastructures in the global economy [1]. Consequently, a comprehensive situational awareness picture is essential in this context.

Research in various domains has addressed methods to enhance maritime situational awareness. Depending on the specific use case, studies may focus on aspects such as cyber security [2]–[4] or sensor data fusion [5], [6]. Regardless of the use case, important locations of interest, such as ports, are often dispersed over vast distances. These specific sites require detailed observation, while simultaneously considering larger areas that need to be monitored as a whole, albeit with less focus on individual details.

To address this challenge, operators often need to constantly switch between higher resolution, zoomed-in views of specific locations and a broader view of the entire area. Alternatively, they may keep multiple viewports open simultaneously. However, it would be advantageous to have a situational awareness picture that offers a focused view of regional areas, such as harbours, while also providing a clear overview of the surrounding water areas, such as seas. This would eliminate the need for users to switch between different views or

maps, enabling them to maintain their situational awareness seamlessly.

In previous studies, magnifying effects for map applications have been presented to cater to diverse use cases [7]–[10]. However, these solutions are typically dependent on the client displaying the data, making it challenging to adapt them to existing situational awareness picture frameworks. In contrast, this work proposes focusing on the middleware while keeping the interfaces unchanged. Consequently, neither the server nor the client needs to be modified to incorporate the shift effect for receiving map material.

II. MOTIVATION

Our approach aims to enhance the observability of the maritime situation by incorporating maps that depict both dispersed local infrastructures, such as ports, and larger areas, such as seas. To achieve this goal, we utilize raster tiles that employ a magnifier effect to showcase detailed views of interesting areas on the map, while still considering the broader regions.

To implement the distortion effect on different base maps using predefined parameters, we require a mechanism that can transmit the distortion information independently from the map material. To fulfil this requirement, we generate shift tiles based on the idea of Normal Maps which are commonly used in computer graphics. These tiles can be seamlessly applied to existing base maps and, by providing them in slippy tile format, can be accessed and, if necessary, further processed by existing applications.

III. PRELIMINARIES

Slippy map tiles are essentially a naming convention for map tiles. The earth is divided into squared tiles and each tile is addressed with the variables x , y and z . The rows and columns are represented by x and y and z represents the zoom factor, meaning that implicitly, a quad tree is created, with the zoom factor defining the depth at which to look for a map tile and x and y specifying the node to look for – meaning that there are 2^z available coordinates for each zoom level z . With the lowest zoom factor 0, the whole earth is mapped with only one tile. Each time the zoom factor is increased by one, the

current tile is divided into four parts and the range of values for x and y is doubled.

This paper also uses the terms x, y and z axis and z layer. The variables x and y describe the coordinates on the x and y axes and start counting at the top left with $0, 0$. Moreover, z describes the current considered zoom level and denotes the value on the z axis. That is to say, z is the z^{th} layer on the z axis.

Each tile, when using raster tiles, measures 256^2 pixels by convention.

A **pixel coordinate** describes with given z the position of a pixel where $\{(x, y) \mid x, y \in \{0, \dots, 256 \cdot 2^z - 1\}\}$ holds. Thus, the pixel coordinate is unique across a z layer throughout all tiles.

The **tile coordinate** describes the position of a tile within a z layer. The domain of a tile coordinate is $\{(x, y) \mid x, y \in \{0, \dots, 2^z - 1\}\}$.

Equation 1 simply computes the distance between two given points p_1 and p_2 . The function $zShiftPix()$ calculates the pixel coordinate for a given position x on the z_s^{th} higher z layer.

$$\text{dist}(p_1, p_2) := p_2 - p_1 \quad (1)$$

$$zShiftPix(x, z_s) := x \cdot 2^{z_s} \quad (2)$$

A **shift tile** stores shift information in its rgb values. Like a normal map stores the shift of a surface's normal vector in a texture's values, shift tiles store a shift of the surface coordinate itself in their colour values. The red, green and blue values are used to indicate a shift on the x, y and z axes, respectively, with the z value indicating a required shift across zoom levels.

For example: We consider tile $(0, 0, 0)$ and want to process the shift information from pixel p at location $(1, 1)$. The displacement information for the pixel under consideration is $p = (-1, -1, 1)$. First we process the value of the z shift 1. So we want to load the new colour value for our pixel under consideration from a z layer upwards. With Equation 2 we get the pixel coordinate $(2, 2)$ for tile $(0, 0, 1)$. Next, apply the shift information for the x and y axes and get the pixel coordinate $(1, 1)$. So as a result, load into tile $(0, 0, 0)$ at location $(1, 1)$ the pixel from tile $0, 0, 1$ from location $(1, 1)$.

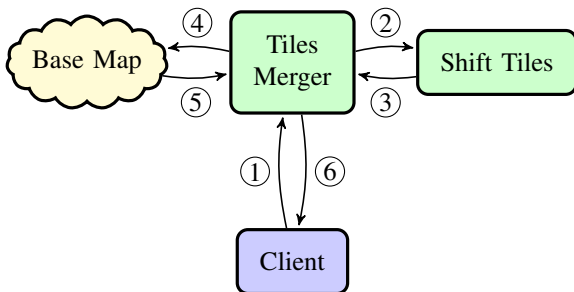


Figure 1. The architecture for a service generating tiles with a magnifying glass effect.

IV. ARCHITECTURE

The fundamental design principle of our architecture is to ensure its independence from both the client and the server, allowing users to remain on their respective platforms while benefiting from non-linearly scaled maps.

All modules within the architecture communicate their requests using rasterised slippy map tiles. This approach offers the advantage of enabling any client to access the map tiles with either the magnifier effect or the shift tiles at any given time.

The architecture, as depicted in Figure 1, comprises four main components: the client, the tiles merger, the base map tile source (which can be any arbitrary source), and the shift tiles module.

The process for requesting map tiles with a magnification effect unfolds as follows:

- 1) The client initiates a request (Step ①) for a specific tile. The tiles merger, which includes information about both the base map and the shift tiles to be used, receives this modified request. This alteration is the only change required for the client, as it remains otherwise unaffected.
- 2) The tiles merger requests the shift tiles module ② for the tile with the shift information and gets it back accordingly ③.
- 3) The tiles merger loads all the required tiles from the base map service ④ and ⑤ based on the shift information provided by the shift tile. It's worth noting that due to the shifting, this may involve tiles from various zoom levels that are necessary for the final tile.
- 4) Next, the tiles merger extracts the pixels from the loaded source map tiles and positions them correctly within the requested tile, as determined by the shift tile.
- 5) Finally, the client receives the requested tile with the magnifier effect ⑥.

V. TILES MERGER

Figure 2 depicts how the tiles merger loads the data¹ from higher z layers if the shift tile contains corresponding information. On the left side you can see the centre of the magnifier effect c and the radius (the highlighted area) to be loaded with data from the next higher layer ($z_s = 1$). On the right side you can see the centre point c' and the radius containing the source pixels to be loaded into the area around c . Here the pixels from four different tiles are loaded, because the new centre point c' in the next higher layer is exactly in the corner of a tile, because according to the Slippy Map Tiles scheme a tile is quartered if you increase the zoom level of the map by one.

The pixel coordinates for the source pixels are obtained with Equation 3. The input values are the centre of the magnifier effect c , the pixel coordinates of the currently considered pixel p for which the shift is calculated and the pixel with the shift information from the Shift Tile s . As output you get the source

¹Tiles generated with data from www.openstreetmap.org

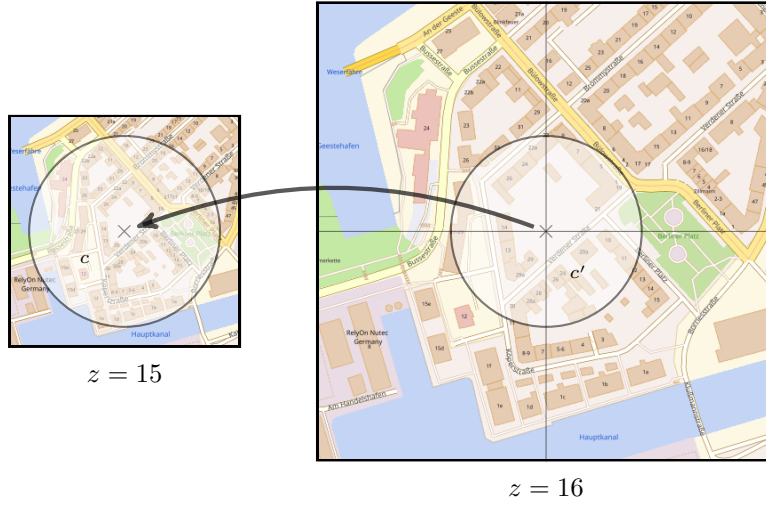


Figure 2. Shows on the left the origin tile, where magnification is desired. The circle on the right marks the area of the next z layer that will be loaded into the area of the circle of the original (left) tile.

pixel coordinate of the pixel that replaces the pixel from the currently considered point p . The shifted pixel coordinate is calculated by first calculating the centre point (c') shifted by $s.z$ with $zShiftPix()$. Next, on top of that, the distance between p and c is added with $dist()$. Thus, the corresponding pixel coordinate of p next to c' is obtained. Lastly, the shift value s is added. This process is carried out for x and y .

$$\begin{aligned} \text{shiftPix}(c, p, s) := \\ & (zShiftPix(c.x, s.z) + \text{dist}(c.x, p.x) + s.x, \quad (3) \\ & zShiftPix(c.y, s.z) + \text{dist}(c.y, p.y) + s.y) \end{aligned}$$

A more detailed procedure of the tiles merger is described in Algorithm 1. First, the center c of the magnifier effect and the shift tile $shTile$ must be set. The result is, depending on the shift tile, a tile with or without a magnifier effect. The input of the function is a tile coordinate x, y, z . First, the related base map tile (line 2) as base source for the resulting tile will be fetched. Next, in line 3 the pixel coordinate bounds are calculated which will be used to iterate on the correct pixel coordinate space. Afterwards, the tiles dimensions are loaded to $width$ and $height$, which are needed in the loop starting at line 5 for computing the pixel coordinate to the coordinate in the given tile. Now, the iteration over the considered tiles pixel coordinates is starting beginning with computing the currently considered coordinate ($curPxlPos$ on line 6), needed for addressing the pixel with the required shift information inside of the current tile. Later $curPxlPos$ is used to put the right pixel into the resulting tile. On line 7 the shift information s is loaded out of the shift tile. With the given shift information s the coordinate of the source pixel $srcPxlPos$ put to the result tile is obtained on line 8. The lines 9 and 10 are computing the tile coordinate $srcTilePos$ of the source tile and fetching the source tile $srcTile$ is done. Next, the source pixel $srcPxl$ is loaded out of the source tile $srcTile$. Here, the modulo operation (mod) is applied on the source pixel

Algorithm 1 Generate magnified tile with given shift tile

Require: center of magnification c

Require: tile with shift information $shTile$

Ensure: a *tile* corresponding to x, y and z

```

1: function GET_MAGNIFIED_TILE( $x, y, z$ )
2:    $tile \leftarrow \text{FETCH\_BASE\_TILE}(x, y, z)$ 
3:    $area \leftarrow \text{PIXEL\_BOUND\_OF\_TILE}(x, y)$ 
4:    $(width, height) \leftarrow tile.dimensions()$ 
5:   for all  $(px, py) \in \{area.left, \dots, area.right \times$ 
       $area.top, \dots, area.bottom\}$  do
6:      $curPxlPos \leftarrow (px \bmod width, py \bmod height)$ 
7:      $s \leftarrow shTile.get\_pixel(curPxlPos)$ 
8:      $srcPxlPos \leftarrow \text{shiftPix}(c, (px, py), s)$ 
9:      $srcTilePos \leftarrow srcPxlPos.to\_tile\_coordinate()$ 
10:     $srcTile \leftarrow \text{FETCH\_BASE\_TILE}(srcTilePos)$ 
11:     $srcPxl \leftarrow srcTile.get\_pixel($ 
       $srcPxlPos.x \bmod width,$ 
       $srcPxlPos.y \bmod height)$ 
12:     $tile.put\_pixel(curPxlPos, srcPxl)$ 
13:   end for
14:   return  $tile$ 
15: end function

```

coordinate $srcPxlPos$, again, because the coordinate inside the considered tile differs from $curPxlPos$ if the source tile is differs from considered tile $tile$ related to x, y and z . Finally, the obtained pixel $srcPxl$ will be put to the resulting tile on line 12 and the iteration continue with the next pixel coordinate. When iteration is done, $\text{GET_MAGNIFIED_TILE}()$ returns the tile with the magnifier effect (if given shift tile contained such data).

VI. SHIFT TILES GENERATION

The versatility of shift tiles lies in their ability to accommodate any required shift, rendering the interfaces of our

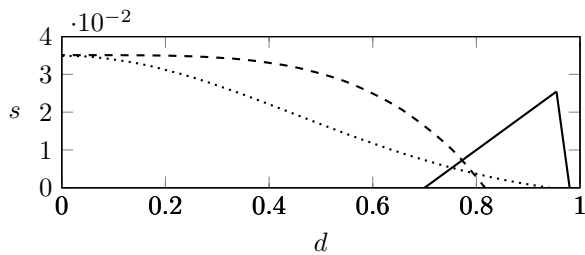


Figure 3. Unified shift functions for x , y and z with distance to c in relation to magnifier effects radius. The horizontal axis denotes the distance d in relation to the radius and the vertical axis represents the shift s of the considered pixel. The solid plot shows the values used for the shift on the x and y axes. Two different functions are used for the shift on the z axis. The dotted and dashed plots show these used functions. While the dotted plot uses Gaussian function the dashed plot uses a fourth grade function for the z shift. By multiplying this value with the unit vector, the shift is obtained.

approach use-case agnostic and adaptable to a wide range of requirements. However, this flexibility naturally leads to the question of *how* these tiles are generated.

In the specific context of magnifying glasses, this work presents a shift map service that does not rely on pre-calculated shift maps. Instead, the service generates the necessary maps procedurally, utilizing parameters provided via the request query string. To illustrate this solution for a particular use case, the generation process will be outlined below.

The shift information for the *shift tiles* is generated differently for the z axis than for the x and y axes. This is shown in Figure 3. A function for the z shift is the Gaussian formula (dotted). This means that the closer the pixel under consideration is to the centre c of the magnifier effect, the larger the shift on the z axis is generated. The shift flattens out towards the edge of the magnifier effect. Thus, at the edge of the magnifier effect and beyond, there is no more z shift. After all, there is no magnification effect beyond the edge of a magnifying glass.

The next function used is a 4th degree function (dashed) like $-d^4 + n$ where d is the distance from the pixel under consideration to the centre and n is how deep the z shift is at maximum. So how much the z shift is around c . If the function gives a negative result, a z shift of 0 is set. If a z shift greater than one is required, this function keeps the rings of the individual z layers narrow, so that the edge of the magnifying effect remains relatively narrow compared to the Gaussian function.

The plot in Figure 3 also shows the proportional shifts for the x and y coordinates using the distance d of the pixel coordinate to the centre c of the magnifying glass effect in relation to each other. For the x and y axes respectively, no shift is generated around the centre c (solid) – instead, the centre of the circle is merely shifted to the next zoom level via a z shift of 1. Beyond a radius of more than 70%, x and y shifts take over, giving a smooth transition from the enlarged centre to the surrounding, original map data. To get the correct shift from the value of Figure 3, it must be multiplied by the unit vector.

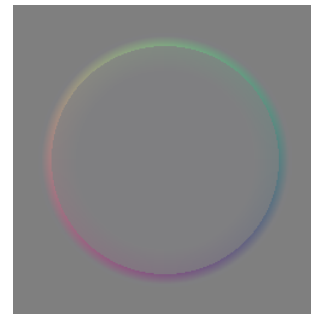


Figure 4. Shift tile with the distortion values of Figure 3. For presentation purposes, the colours have a higher contrast than the real shift tiles to illustrate their values in a range that is visible to the human eye.

An example of a shift tile can be seen in Figure 4. It is easy to see the ring representing the shift on the x and y axes. The colour of the shift values of the z axis, on the other hand, is more subtle – as a value of 1 represents the next higher zoom level (and thus a magnification by a factor of 2), the values tend to be lower, making them harder to see.

VII. RESULTS

Figure 5 shows the resulting maps generated using the proposed shift tiles, showing the fishing port of Bremerhaven. For the map in the middle, the magnifier effect was generated with the Gaussian function. You can clearly see from the wide border that as the distance to the centre increases, the shift value becomes smaller only very slowly (see Figure 3, dotted line). The map on the right, on the other hand, has only a narrow border.

It is easy to see that the magnifier effect with the fourth degree function requires less area for the distortion effects. Thus, there is more space for regular map information on this map than on the map in the middle. As intended, both maps show one area in more detail (high zoom level) without neglecting the rest of the port.

VIII. FUTURE WORK

A very useful tool for this could be another service that offers, for example, the position data of ships using *Automatic Identification System* (AIS) [11] in near real time. With the help of the shift tiles, the coordinates for the pins in the map client would be adjusted, allowing them to be correctly displayed on the map with a magnifying glass effect. In this way, the positions of the ships are correctly displayed on the map despite the distortion effect, without the need for a new client.

REFERENCES

- [1] UNCTAD, *Review of Maritime Transport 2022, Navigating stormy waters*. United Nations, 2022. DOI: [10.18356/9789210021470](https://doi.org/10.18356/9789210021470).
- [2] G. C. Kessler, J. P. Craiger and J. C. Haass, 'A taxonomy framework for maritime cybersecurity: A demonstration using the automatic identification system', *TransNav: International Journal on Marine Navigation and Safety of Sea Transportation*, vol. 12, no. 3, pp. 429–437, 2018.

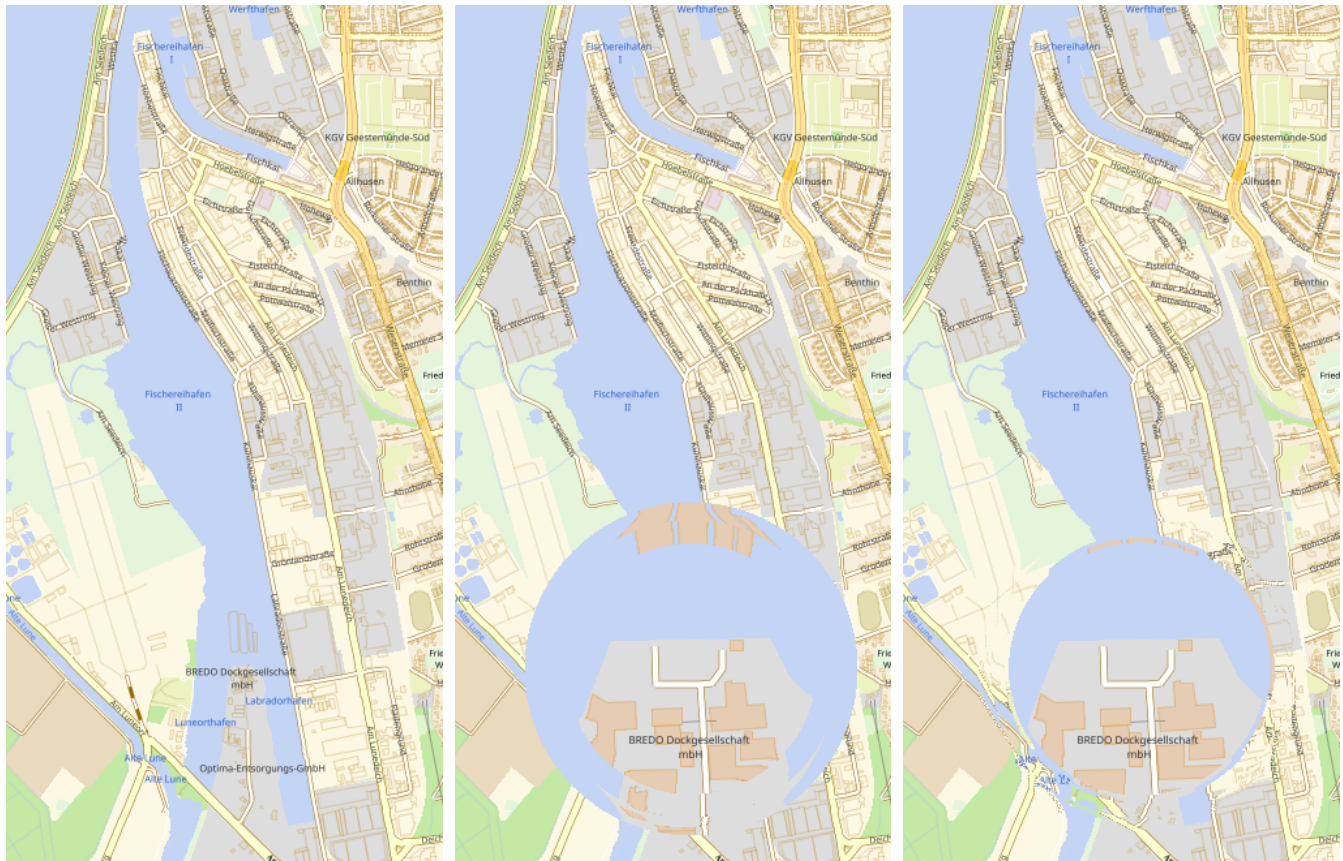


Figure 5. On the left side is the base map without any magnifier effects. The middle shows the result of the map with the magnifier effect using the Gaussian function for the z shift. On the right side is the map with the magnifier effect using the function of the fourth grade.

- [3] O. Jacq, 'Realtime cyberattacks detection, contextual analysis and visualization: Maritime cyber situational awareness for the maritime domain', Ph.D. dissertation, Jan. 2021.
- [4] F. Akpan, G. Bendiab, S. Shiaeles, S. Karamperidis and M. Michaloliakos, 'Cybersecurity challenges in the maritime sector', *Network*, vol. 2, no. 1, pp. 123–138, 2022, ISSN: 2673-8732. DOI: [10.3390/network2010009](https://doi.org/10.3390/network2010009).
- [5] A. C. van den Broek, R. M. Neef, P. Hanckmann, S. P. van Gosliga and D. van Halsema, 'Improving maritime situational awareness by fusing sensor information and intelligence', in *14th International Conference on Information Fusion*, 2011, pp. 1–8.
- [6] L. Snidaro, I. Visentini and K. Bryan, 'Fusing uncertain knowledge and evidence for maritime situational awareness via markov logic networks', *Information Fusion*, vol. 21, pp. 159–172, 2015, ISSN: 1566-2535. DOI: <https://doi.org/10.1016/j.inffus.2013.03.004>.
- [7] L. Harrie, L. Sarjakoski and L. Lehto, 'A variable-scale map for small-display cartography', *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, vol. 34, Jan. 2002.
- [8] F. Jia, X. You, Y. Liu and G. Song, 'Implementation of magnifier effect in multi-scale display of electronic map', in *Geoinformatics 2008 and Joint Conference on GIS and Built Environment: Geo-Simulation and Virtual GIS Environments*, L. Liu, X. Li, K. Liu, X. Zhang and A. Chen, Eds., International Society for Optics and Photonics, vol. 7143, SPIE, 2008, p. 714 339. DOI: [10.1117/12.812650](https://doi.org/10.1117/12.812650).
- [9] R. Zhao, T. Ai and C. Wen, 'A method for generating variable-scale maps for small displays', *ISPRS International Journal of Geo-Information*, vol. 9, no. 4, 2020, ISSN: 2220-9964. DOI: [10.3390/ijgi9040250](https://doi.org/10.3390/ijgi9040250).
- [10] Z. Chen, Y. Shen, X. Lv, Q. Qin and X. Chen, 'Variable-scale visualization of high-density polygonal buildings on a tile map', *ISPRS International Journal of Geo-Information*, vol. 11, no. 10, 2022, ISSN: 2220-9964. DOI: [10.3390/ijgi11100505](https://doi.org/10.3390/ijgi11100505).
- [11] 'M.1371: Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile frequency band', en, International Telecommunication Union (ITU), Geneva, CH, Standard M.1371-5, 2014.