Université Clermont Auvergne

Licence 3 Informatique

Réseaux 2

Projet: Graphical Communication Protocol (Barcode System)

Auteur:
Engelbrecht Jérôme

Summary

The goal of this project was to design a custom graphical communication protocol inspired by QR codes, yet fully original. The protocol transmits a message visually using a linear colored barcode system. This system provides: (1) data storage, (2) error detection and correction, and (3) robustness against noise and orientation issues. The accompanying program can encode text into an image and decode the image back into the original message, even after simulated noise and distortions.

The program can be tested by running the `main.py` file in Python. Several prompts will appear in the console; to use default values, simply press Enter at each step. You can then enter your message in the final prompt. Two images are generated during the process: `barcode.png` (the original generated code, with applied scaling, noise, and rotation) and `barcode_formatted.png` (the image as reconstructed by the decoding algorithm). To customize the logo, replace the file `logo.jpg` with any image of your choice using the same filename.

Contents

1 Project Architecture

The system is structured as a pipeline:

1. The user provides a string message.
2. The `encoder.py` module encodes this data into a byte array with Reed-Solomon error correction.
3. The `barcode_drawer.py` module generates an image from this array, adding visual patterns and optional noise.
4. The system can simulate real-world distortions (rotation, scale, noise).
5. The `utils.py` module reads back the image, corrects orientation, extracts data, and validates the decoded message.

2 Design Decisions

2.1 Linear Barcode Format

We intentionally chose a 1D colored barcode rather than a 2D matrix (like QR codes). This makes it easier to read when partially occluded, which is very important when choosing a color based bytes encoding. To sample the color of a byte we average a large portion of color region, leading to a greater accuracy.

2.2 Fixed Information Zones

The barcode is divided into well-defined zones:

- Finder patterns for orientation and scale detection
- Error Correction Level indicator
- Logo insertion (optional, does not overwrite data)
- Data section (main payload)
- Padding if data does not fully occupy expected size
- End finder pattern for termination detection

2.3 Error Correction

Reed-Solomon codes (`reedsolo` library) were used, as they are the gold standard in many real-world graphical communication protocols. The user can configure the error correction strength (0–255).

2.4 Noise Resistance

The protocol was designed to resist:

- Image rotation (finder patterns detect orientation)
- Resizing and distortion (scale detected from bar width)
- Pixel-level color noise (high tolerance in decoding)

3 Protocol Schema Explanation

Figure 1: Structure of the graphical communication protocol

The protocol is structured as a linear sequence of colored vertical bars, each bar representing either metadata, control, or data. Each bar covers a fixed width (`barWidth`) and full image height. The structure is as follows:

3.1 Finder Patterns (bars 0 and 2)

Two solid black bars at positions 0 and 2 act as orientation markers. They allow the decoder to:

- Detect image rotation: the algorithm checks for the black-colored–color–black pattern at any rotation (0, 90, 180, 270 degrees).
- Compute the bar width: by measuring distance between the first two black bars.

This is implemented in `utils.format_image()` by scanning the middle row of the image.

3.2 Error Correction Level (bar 1)

Located between the two initial finder patterns, this single colored bar encodes the error correction level (0–255). The value is scaled into HSV hue:

```
hue = (byte / 255.0) * 360
rgb = hsv_to_rgb(hue / 360.0, 1.0, 1.0)
```

During decoding, the average color is converted back to estimate the ECC level, which determines how many Reed-Solomon ECC symbols are expected.

3.3 Logo Insert (bars after 3)

Immediately after the first three bars, a reserved space is allocated for an optional embedded logo.

- The logo image (`logo.jpg`) is resized and inserted without altering any data.
- The space is calculated as:

```
logo_width_in_bars = max(1, round(logo.width / barWidth))
```

- It visually personalizes the barcode without modifying any payload or control data.

The decoder skips these bars since the gap size can be re-computed using the known barHeight/barWidth ratio.

3.4 Data Bytes

Following the logo area, the remaining bars encode the message data. Each byte is mapped to a single bar using HSV hue values:

- The system maps values 0–255 to the full hue circle.
- This guarantees full coverage of the RGB color space and ensures distinguishability under printing or camera capture.
- Color sampling uses the full bar area (all pixels in a bar) for maximal accuracy.

The data is first processed by Reed-Solomon encoding:

```
encoded = RSCodec(error_correction_level).encode(data)
```

so that any small errors can be corrected at decoding.

3.5 Padding

If the data and ECC values do not fill the remaining array size, zeroes (0x00) are appended:

```
if len(encoded) < array_size - 4:
    encoded += bytearray(array_size - 4 - len(encoded))
```

Padding ensures that every barcode has a fixed size, simplifying parsing and improving reliability.

3.6 Code End Finder (last bar)

The last bar of the barcode is always a solid black bar. This allows the reader to:

- Easily detect the end of the barcode.
- Reject invalid scans where an end marker is missing.

It mirrors the first black finder and increases tolerance to partial image cuts.

4 Conclusion

This protocol achieves full transmission of short text messages through images, with high robustness to print distortions and rotation. The design is modular and customizable by adjusting:

- Bar size
- Error correction strength
- Noise tolerance level
- Logo size

Future improvements could include a stronger image formatter to handle more complicated rotations and angles than the ones imposed in this project.