Prometheus

A Smartphone-Based Screen-to-Camera Communication System

PAN Weiheng

Aug 20, 2020

Contributions

Contributions

- First open-source iOS app for screen-to-camera (S2C) communication

Contributions

- First open-source iOS app for screen-to-camera (S2C) communication
    - A starting point for other researchers

Contributions

- First open-source iOS app for screen-to-camera (S2C) communication
    - A starting point for other researchers
- Alternating code display mode

Contributions

- First open-source iOS app for screen-to-camera (S2C) communication
    - A starting point for other researchers
- Alternating code display mode
    - Nearly doubles the throughput

Contributions

- First open-source iOS app for screen-to-camera (S2C) communication
    - A starting point for other researchers
- Alternating code display mode
    - Nearly doubles the throughput
    - 204 Kbps without any packet losses

Contributions

- First open-source iOS app for screen-to-camera (S2C) communication
  - A starting point for other researchers
- Alternating code display mode
  - Nearly doubles the throughput
  - 204 Kbps without any packet losses
- Nested code display mode

Contributions

- First open-source iOS app for screen-to-camera (S2C) communication
  - A starting point for other researchers
- Alternating code display mode
  - Nearly doubles the throughput
  - 204 Kbps without any packet losses
- Nested code display mode
  - First to utilize dual camera system in S2C

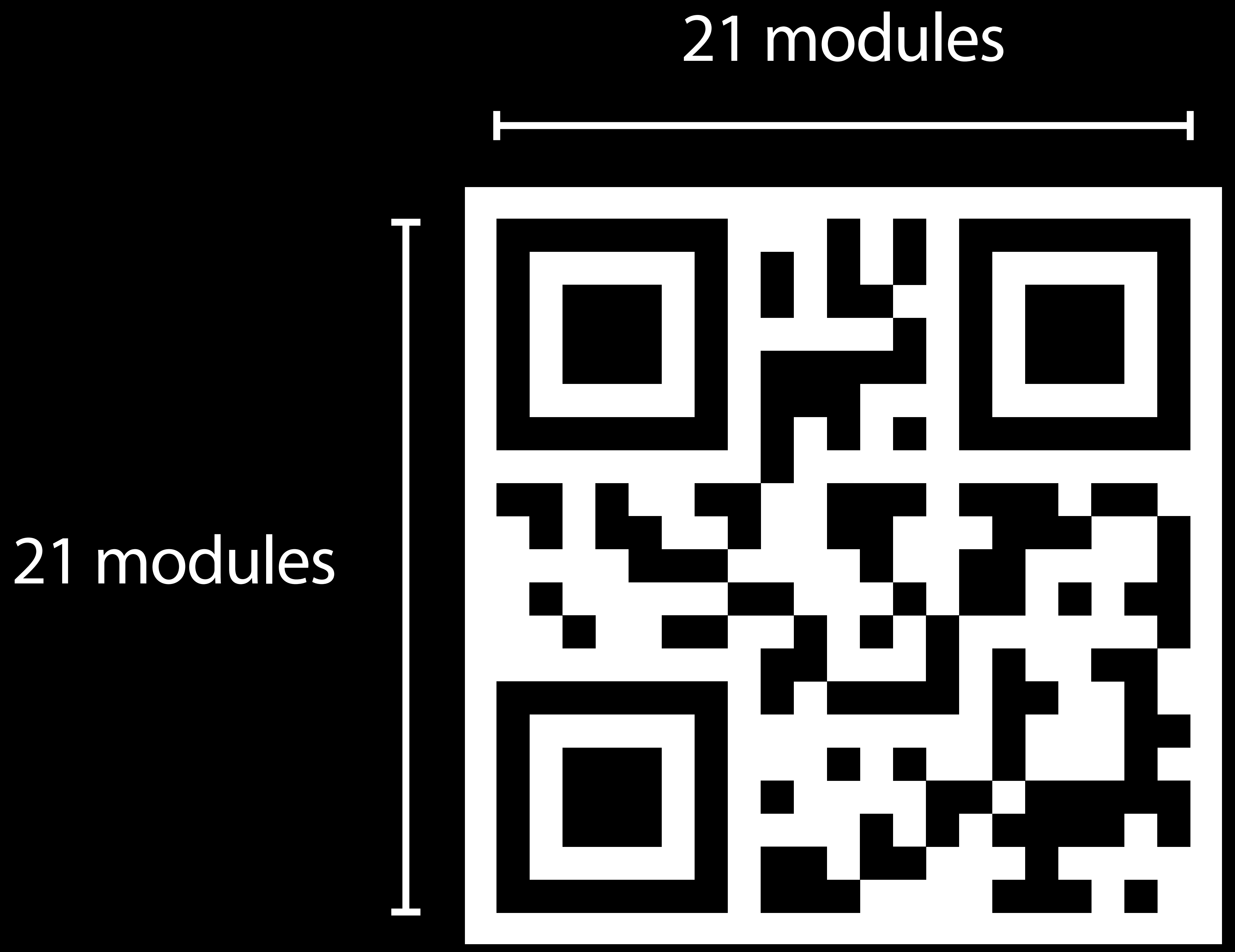
Contributions

- First open-source iOS app for screen-to-camera (S2C) communication
    - A starting point for other researchers
- Alternating code display mode
    - Nearly doubles the throughput
    - 204 Kbps without any packet losses
- Nested code display mode
    - First to utilize dual camera system in S2C
    - ~20% improvement in throughput

Contributions

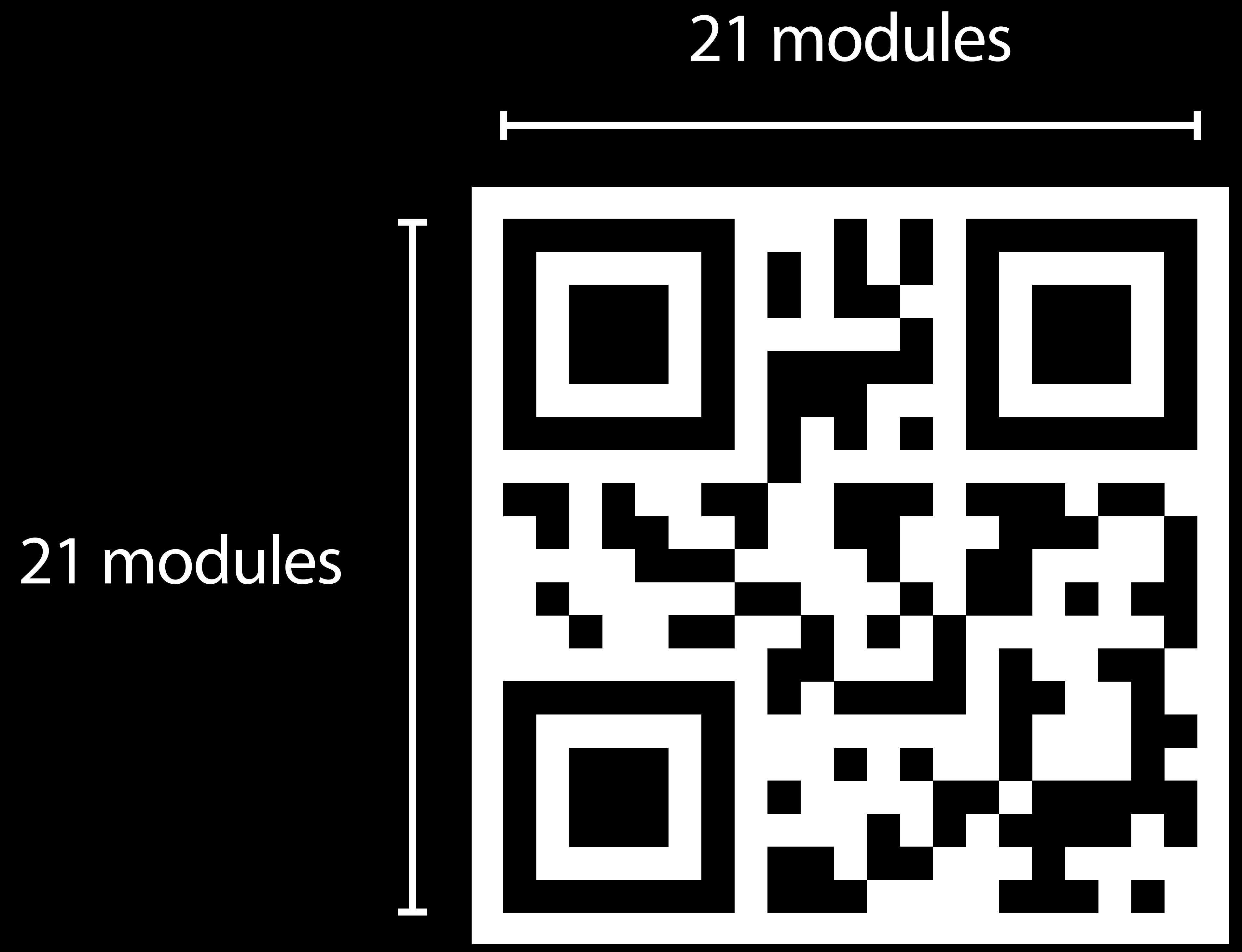
- First open-source iOS app for screen-to-camera (S2C) communication
  - A starting point for other researchers
- Alternating code display mode
  - Nearly doubles the throughput
  - 204 Kbps without any packet losses
- Nested code display mode
  - First to utilize dual camera system in S2C
  - ~20% improvement in throughput
- First (asymmetrical) duplex S2C communication system with retransmissions

QR Code Terminologies



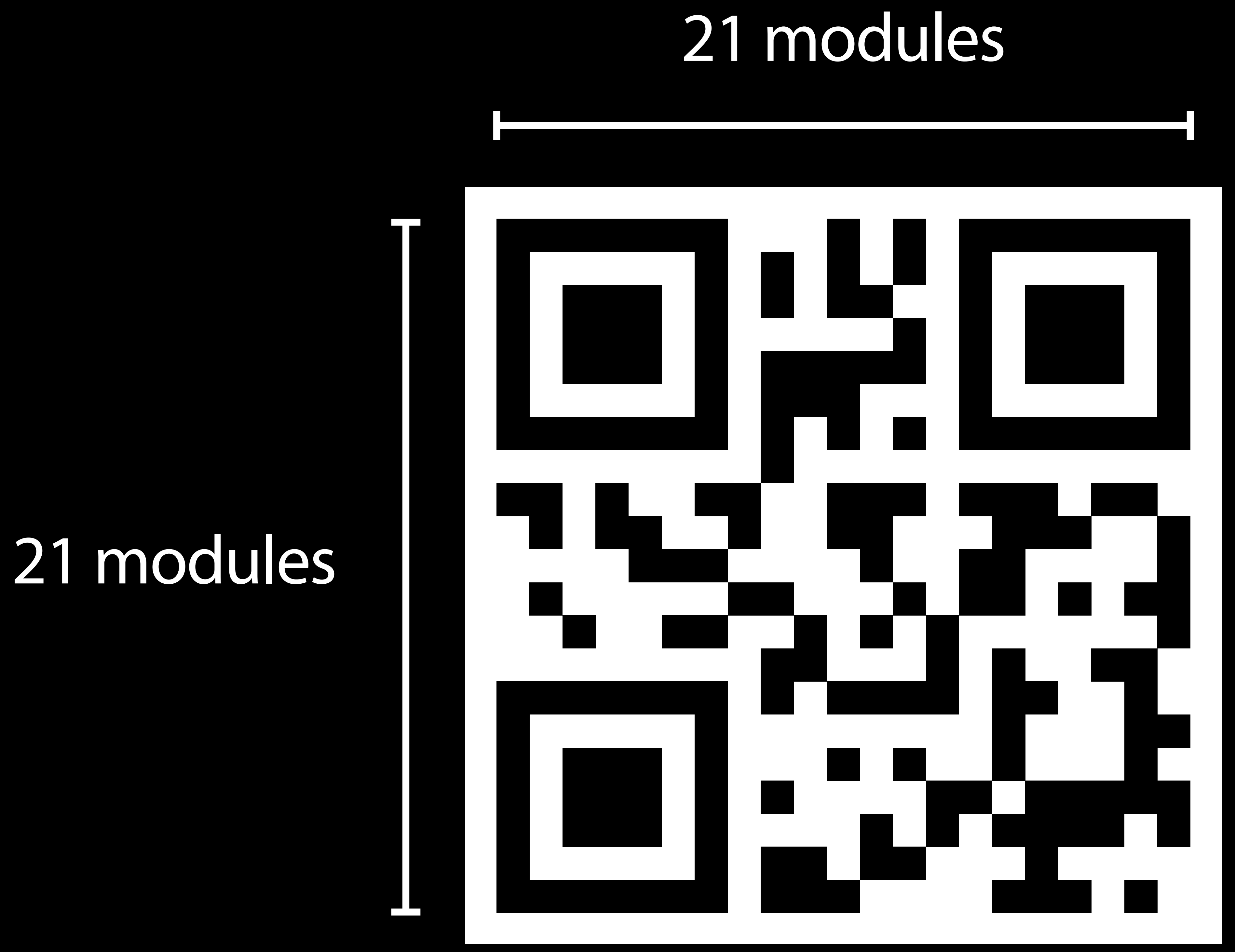
QR Code Terminologies

- Module

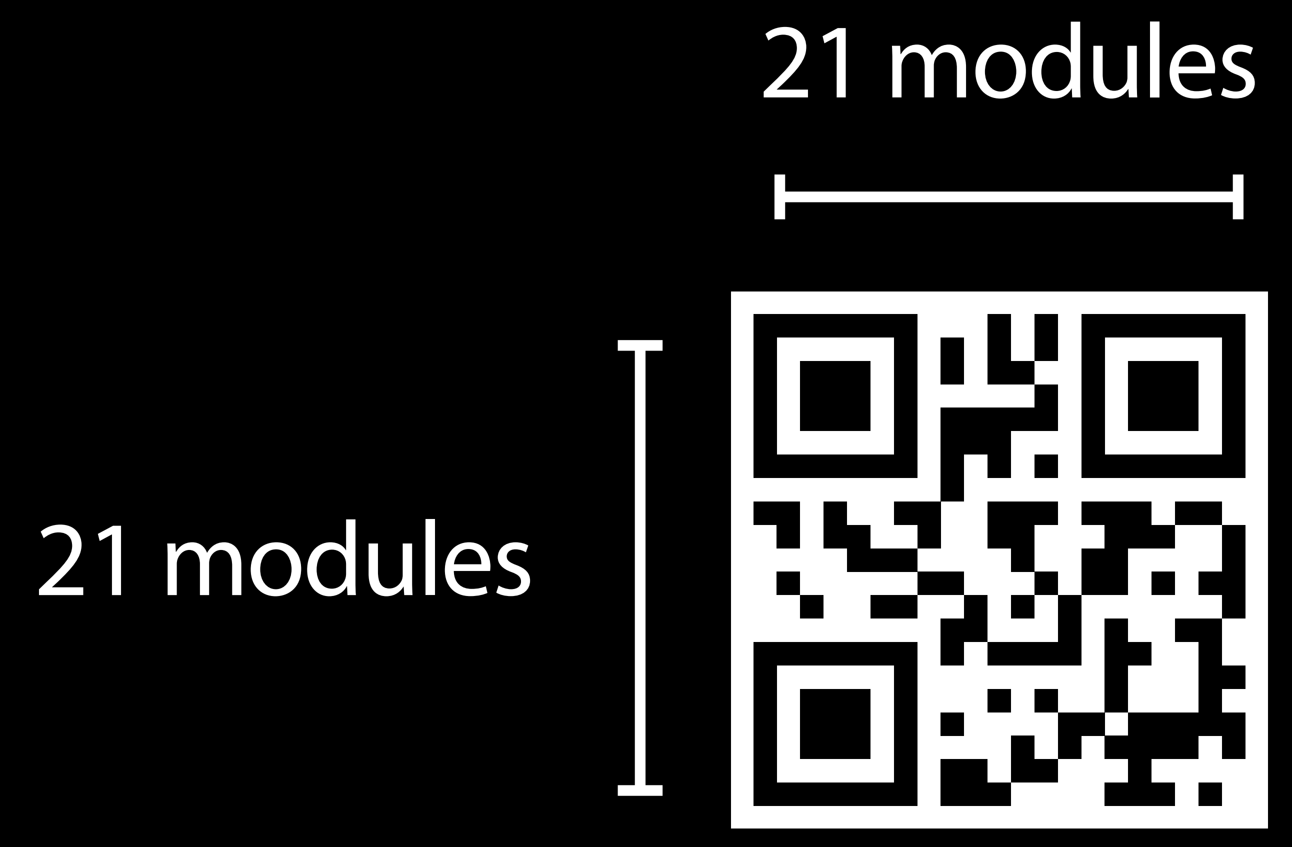


QR Code Terminologies

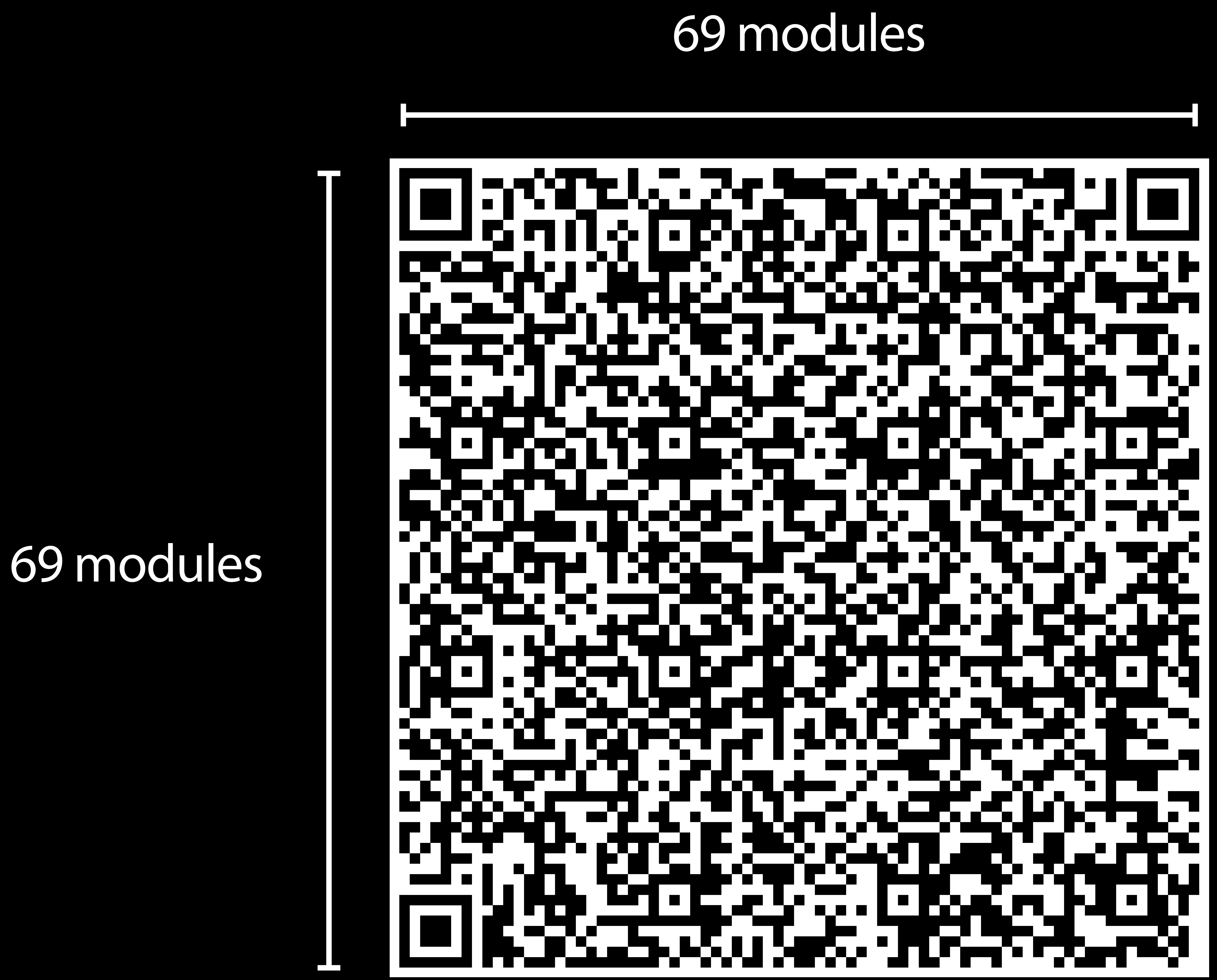
- Module
- Version



QR Code Terminologies



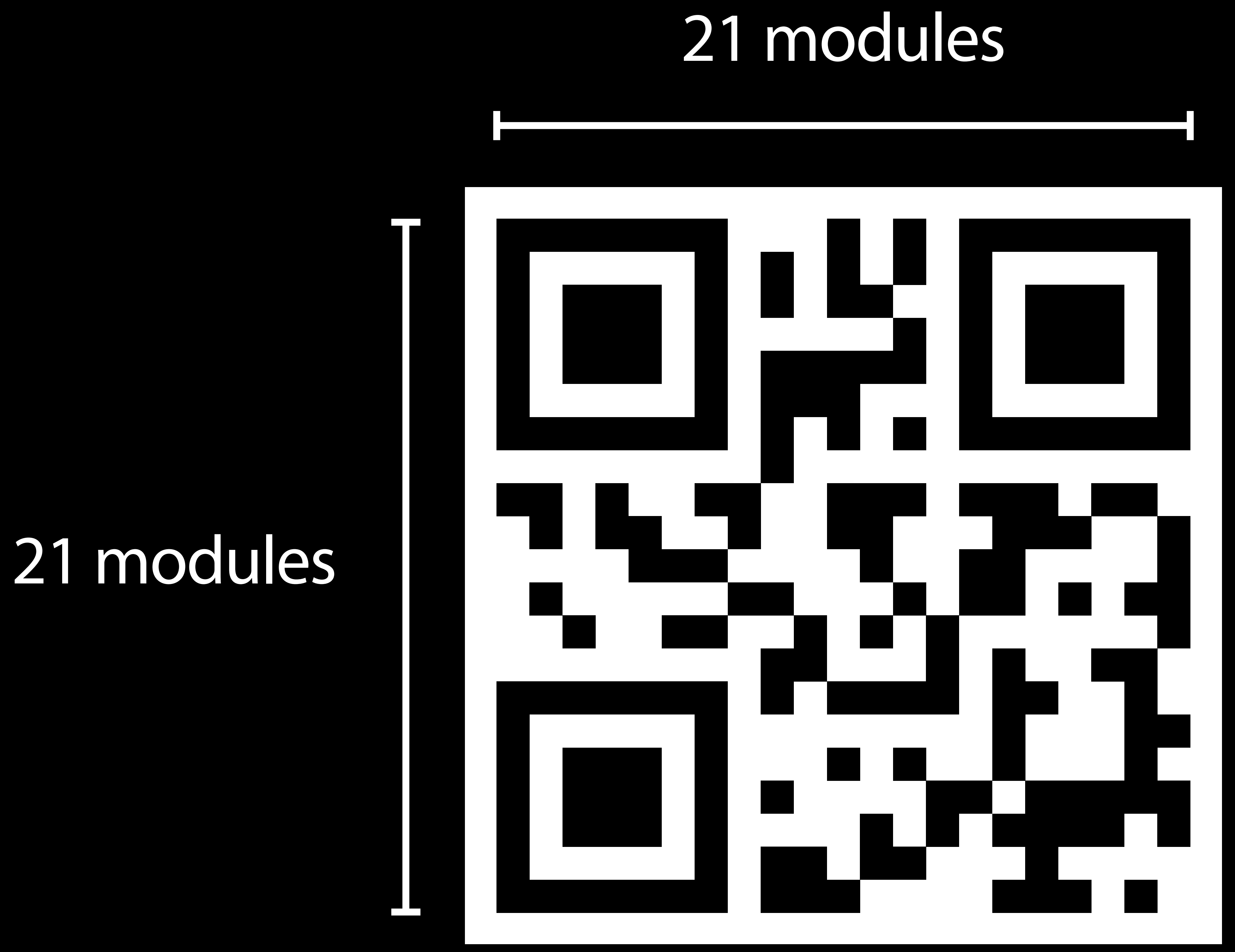
Version 1



Version 13

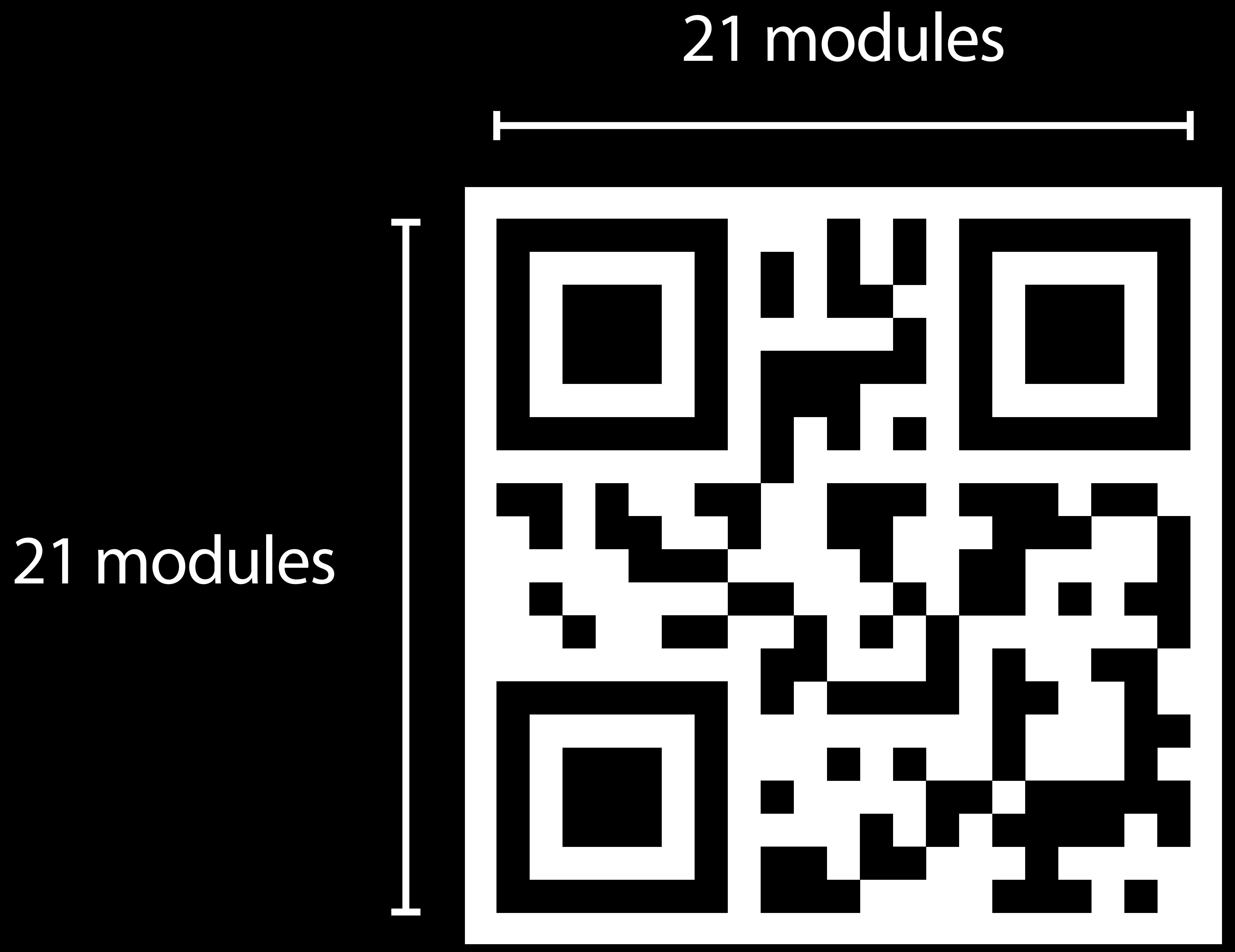
QR Code Terminologies

- Module
- Version

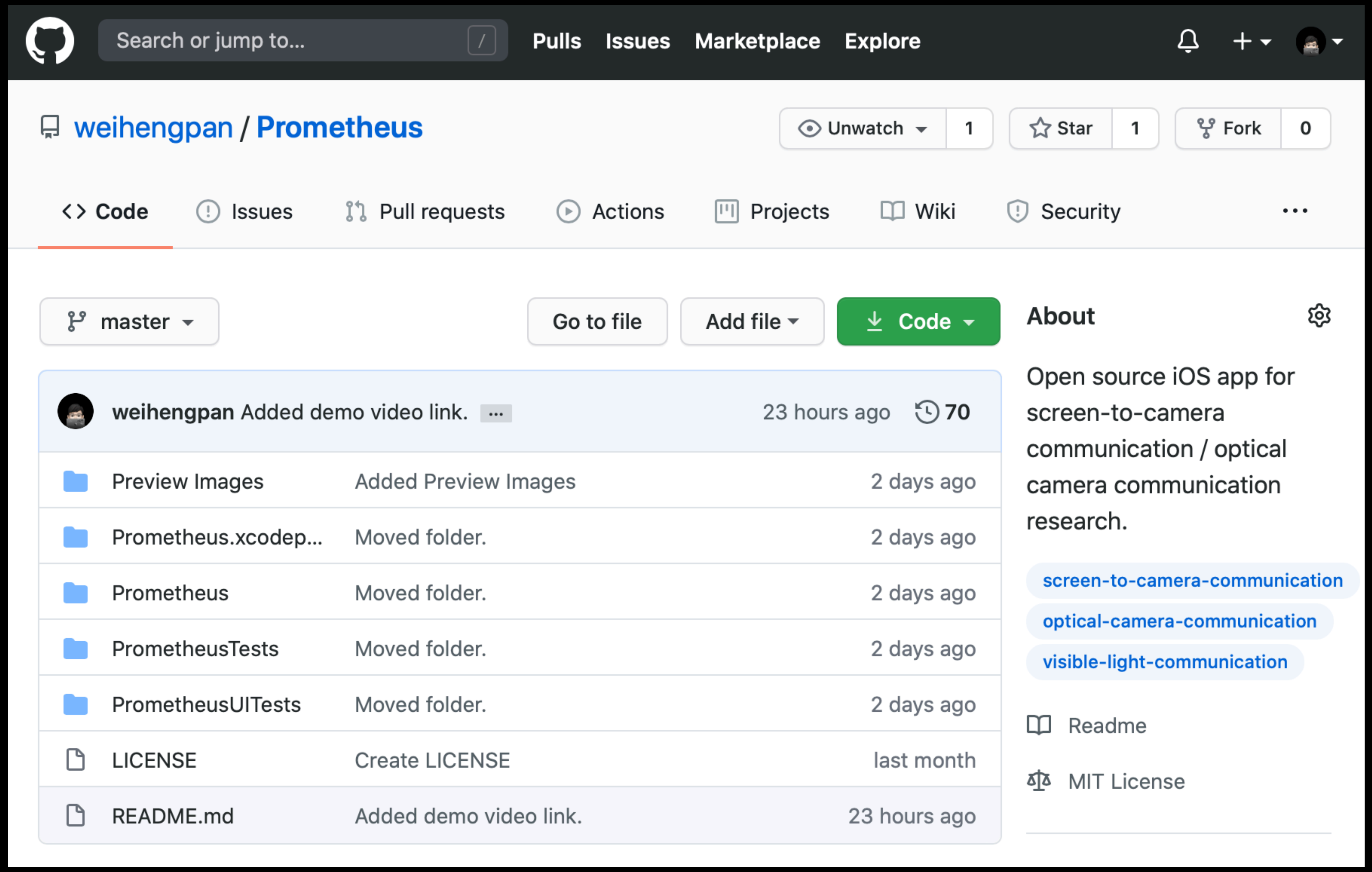


QR Code Terminologies

- Module
- Version
- Error Correction Level
    - Low (7%)
    - Medium (15%)
    - Quartile (25%)
    - High (30%)



Prometheus



weihengpan / Prometheus

weihengpan Added demo video link. 23 hours ago 70

Preview Images	Added Preview Images	2 days ago
Prometheus.xcodep...	Moved folder.	2 days ago
Prometheus	Moved folder.	2 days ago
PrometheusTests	Moved folder.	2 days ago
PrometheusUITests	Moved folder.	2 days ago
LICENSE	Create LICENSE	last month
README.md	Added demo video link.	23 hours ago

About

Open source iOS app for screen-to-camera communication / optical camera communication research.

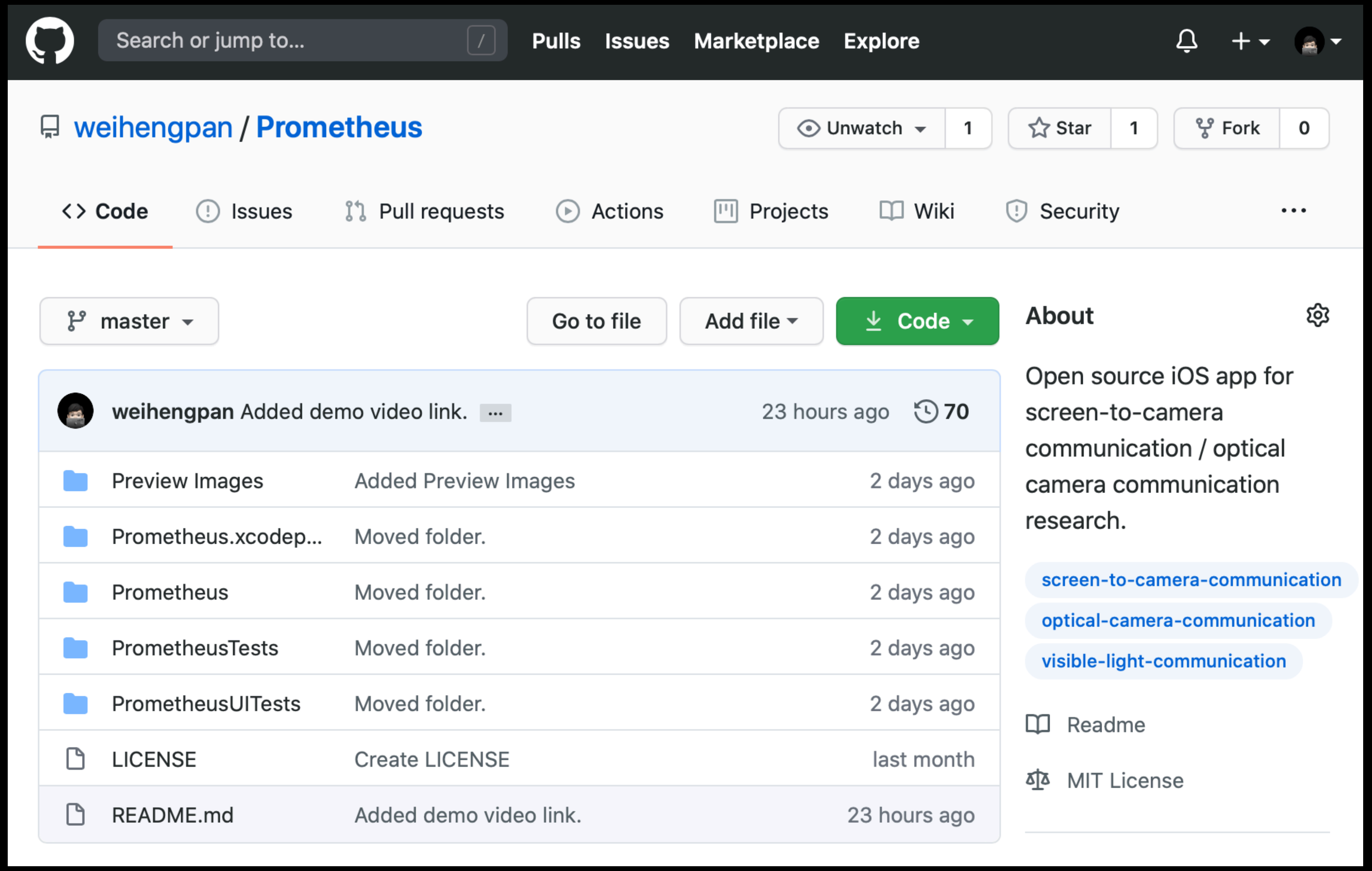
screen-to-camera-communication
optical-camera-communication
visible-light-communication

Readme

MIT License

Prometheus

- About 4,000 lines of Swift code



weihengpan Added demo video link. 23 hours ago 70

Preview Images	Added Preview Images	2 days ago
Prometheus.xcodep...	Moved folder.	2 days ago
Prometheus	Moved folder.	2 days ago
PrometheusTests	Moved folder.	2 days ago
PrometheusUITests	Moved folder.	2 days ago
LICENSE	Create LICENSE	last month
README.md	Added demo video link.	23 hours ago

About

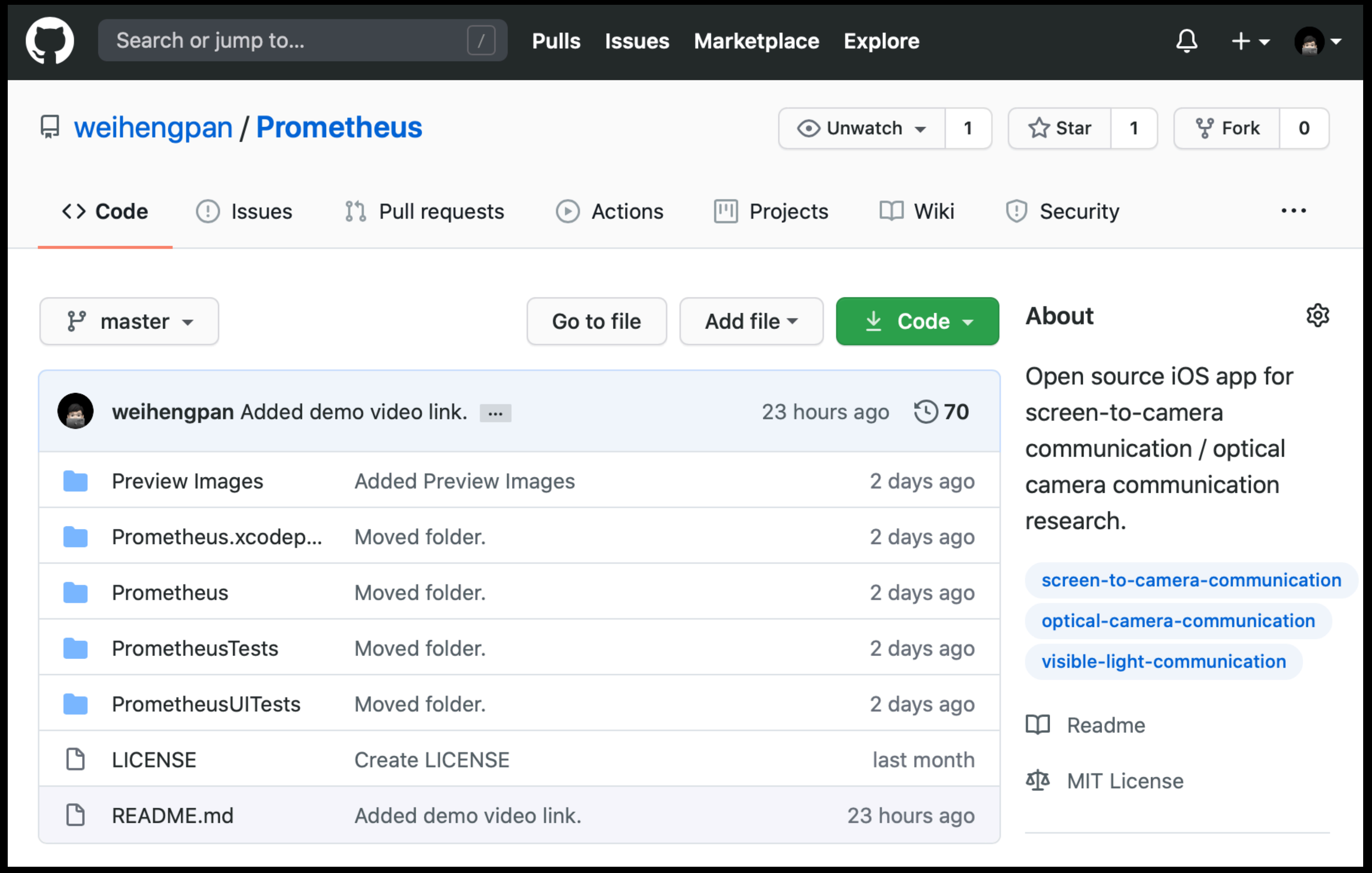
Open source iOS app for screen-to-camera communication / optical camera communication research.

screen-to-camera-communication
optical-camera-communication
visible-light-communication

Readme
MIT License

Prometheus

- About 4,000 lines of Swift code
- Requires iOS 13.0



weihengpan / Prometheus

weihengpan Added demo video link. 23 hours ago 70

Preview Images	Added Preview Images	2 days ago
Prometheus.xcodep...	Moved folder.	2 days ago
Prometheus	Moved folder.	2 days ago
PrometheusTests	Moved folder.	2 days ago
PrometheusUITests	Moved folder.	2 days ago
LICENSE	Create LICENSE	last month
README.md	Added demo video link.	23 hours ago

About

Open source iOS app for screen-to-camera communication / optical camera communication research.

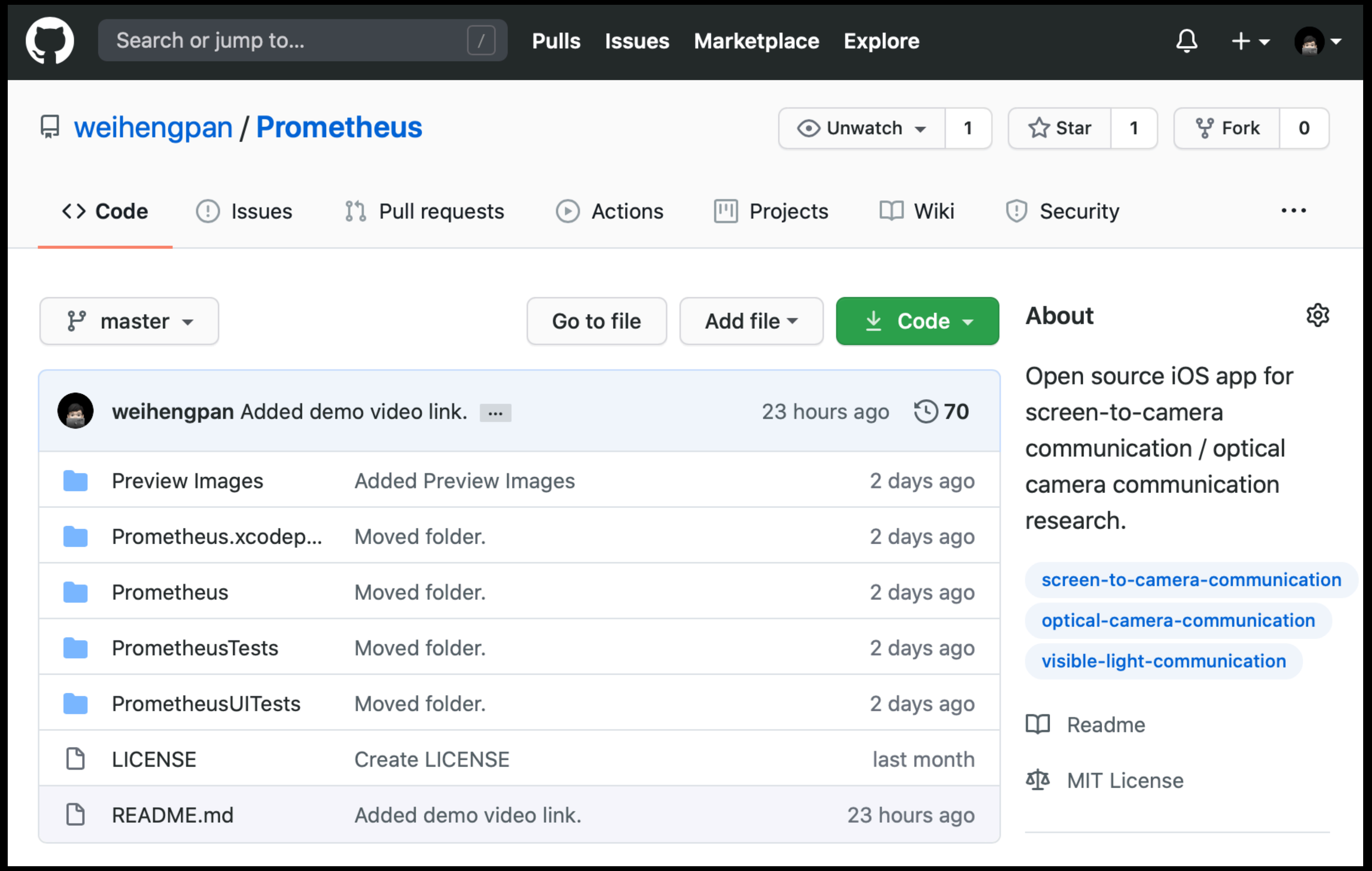
screen-to-camera-communication
optical-camera-communication
visible-light-communication

Readme

MIT License

Prometheus

- About 4,000 lines of Swift code
- Requires iOS 13.0
- Open source under MIT License



weihengpan Added demo video link. 23 hours ago 70

Preview Images	Added Preview Images	2 days ago
Prometheus.xcodep...	Moved folder.	2 days ago
Prometheus	Moved folder.	2 days ago
PrometheusTests	Moved folder.	2 days ago
PrometheusUITests	Moved folder.	2 days ago
LICENSE	Create LICENSE	last month
README.md	Added demo video link.	23 hours ago

About

Open source iOS app for screen-to-camera communication / optical camera communication research.

screen-to-camera-communication
optical-camera-communication
visible-light-communication

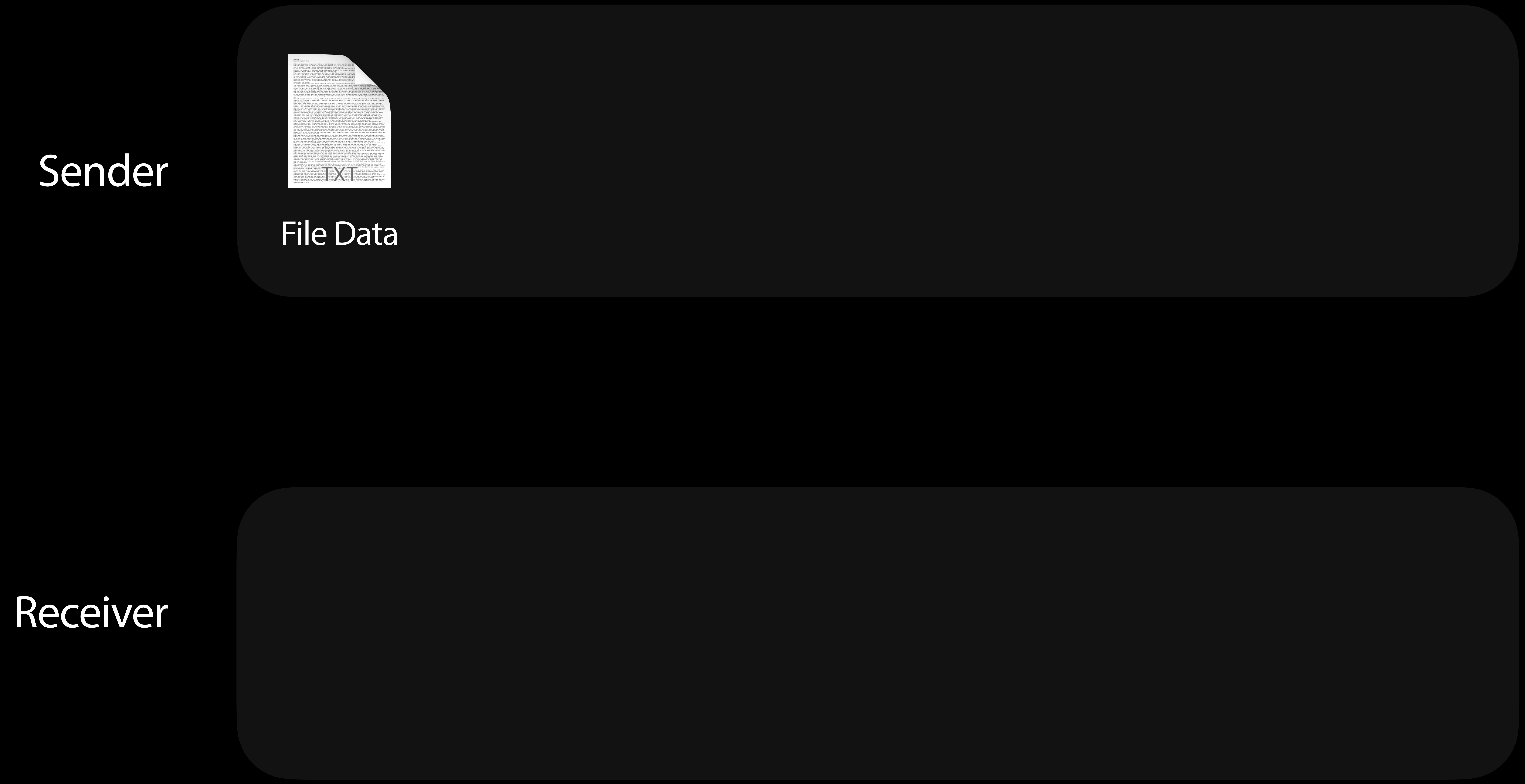
Readme
MIT License

System Architecture

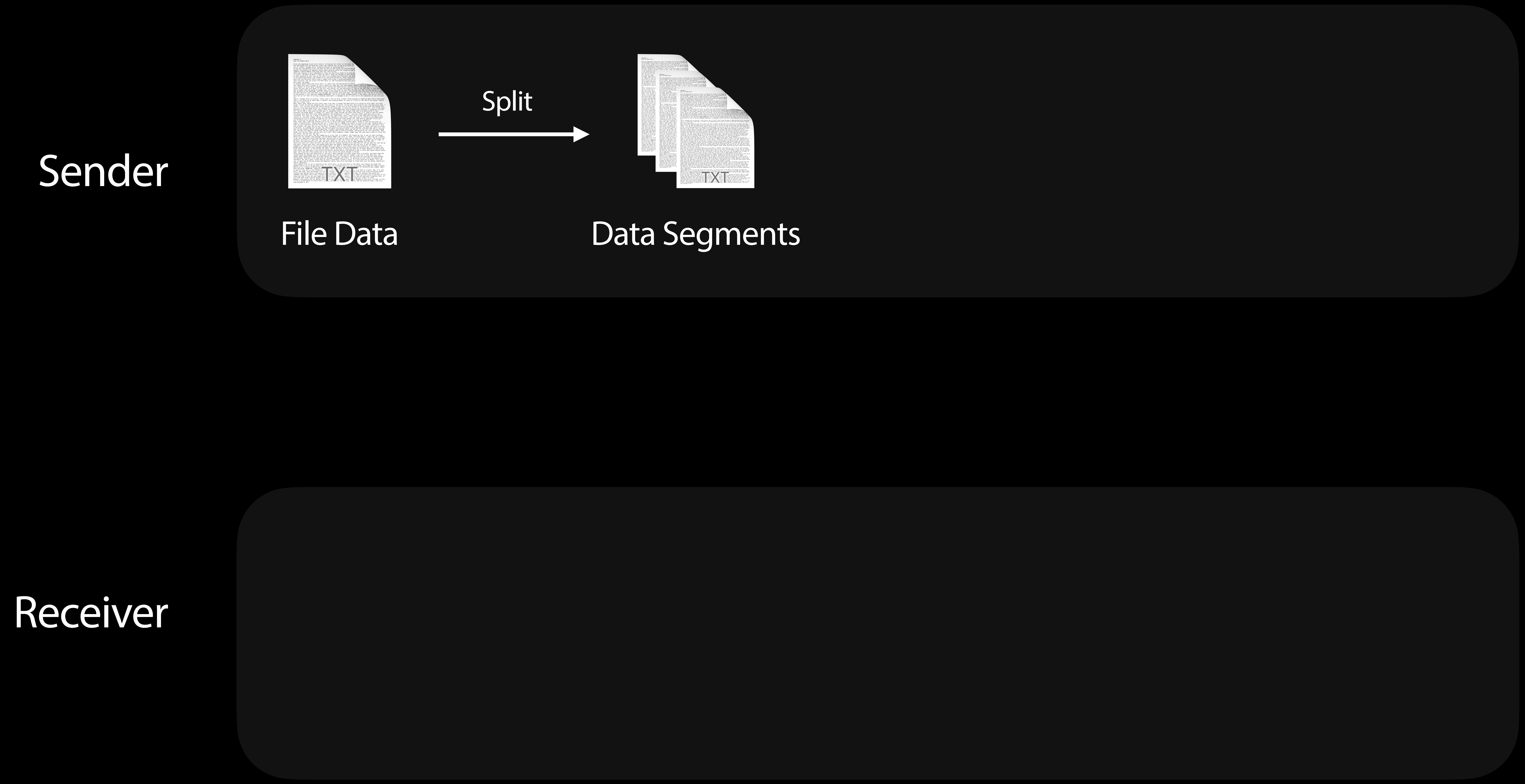
Sender

Receiver

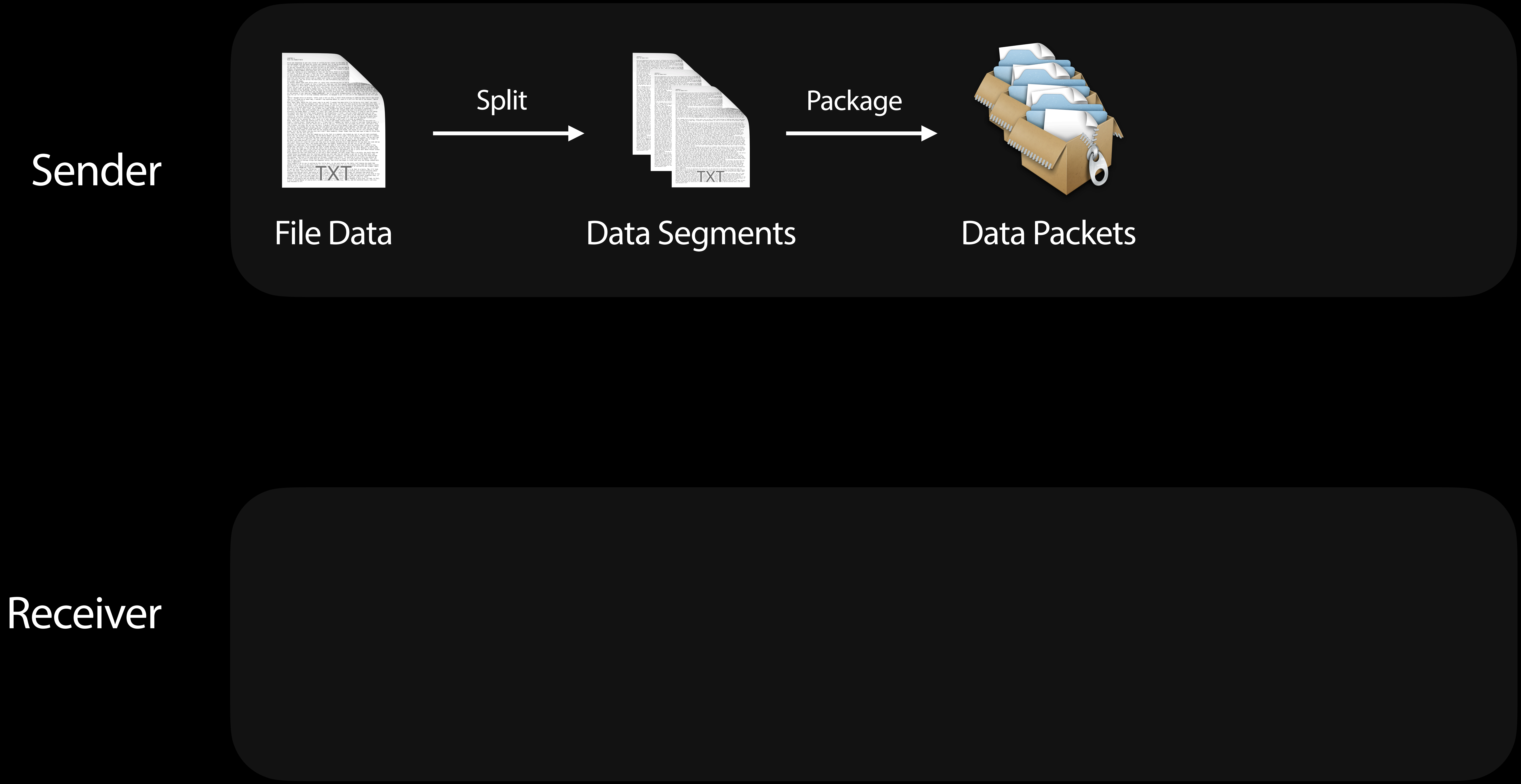
System Architecture



System Architecture

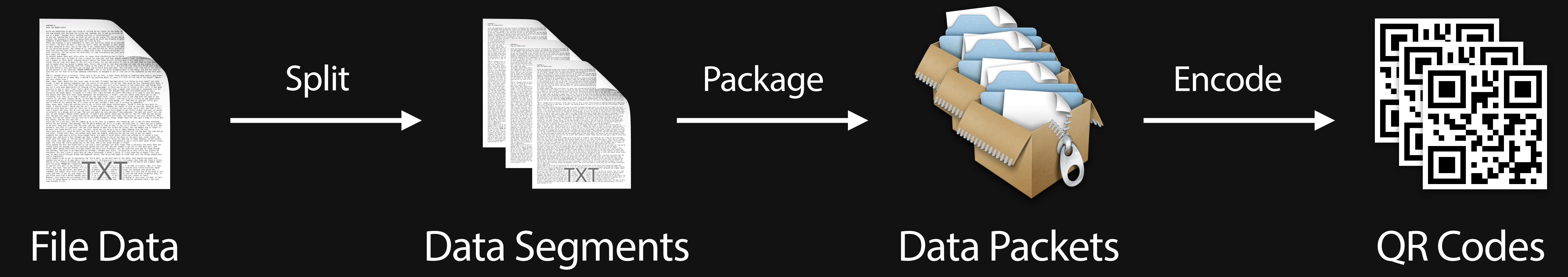


System Architecture



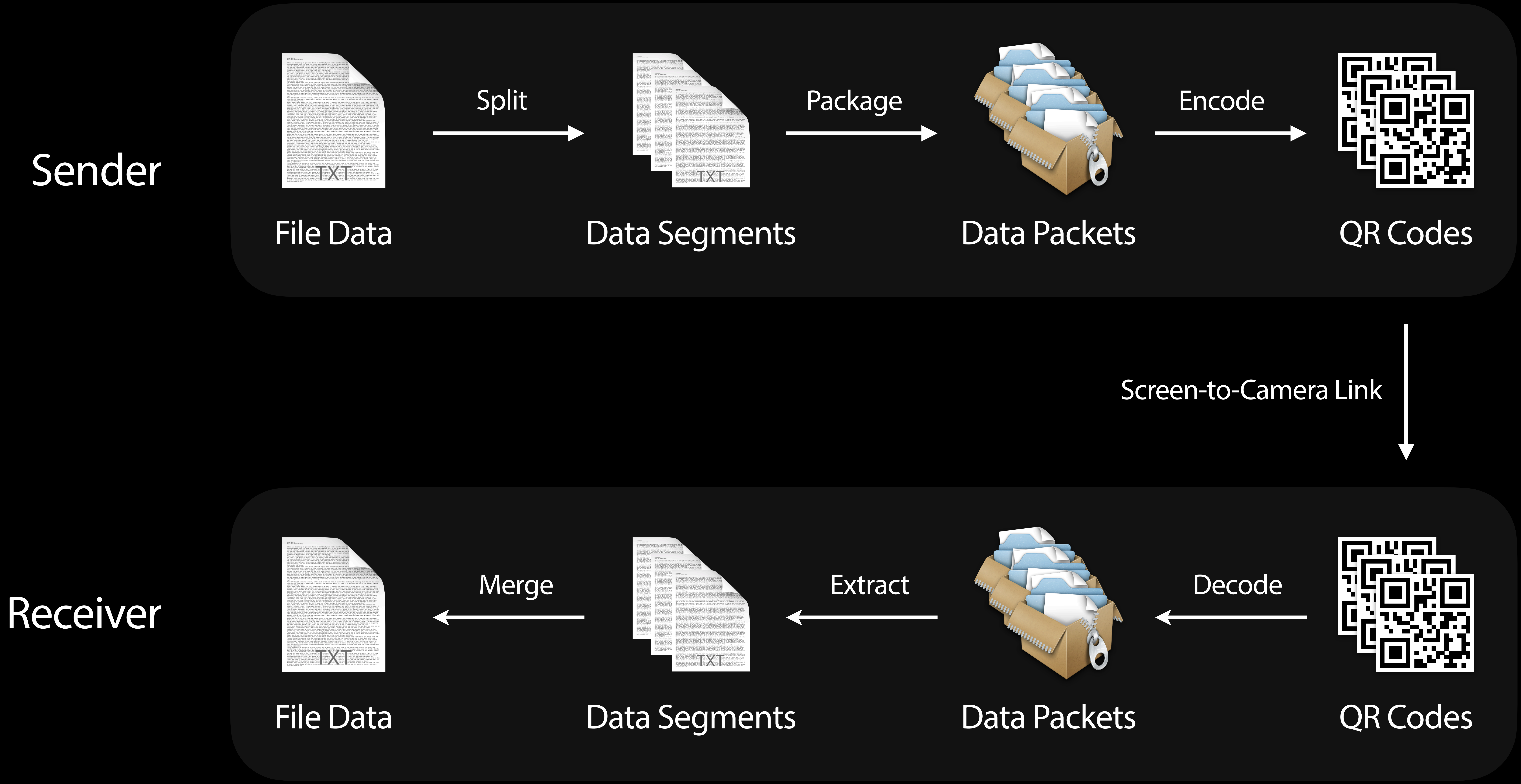
System Architecture

Sender



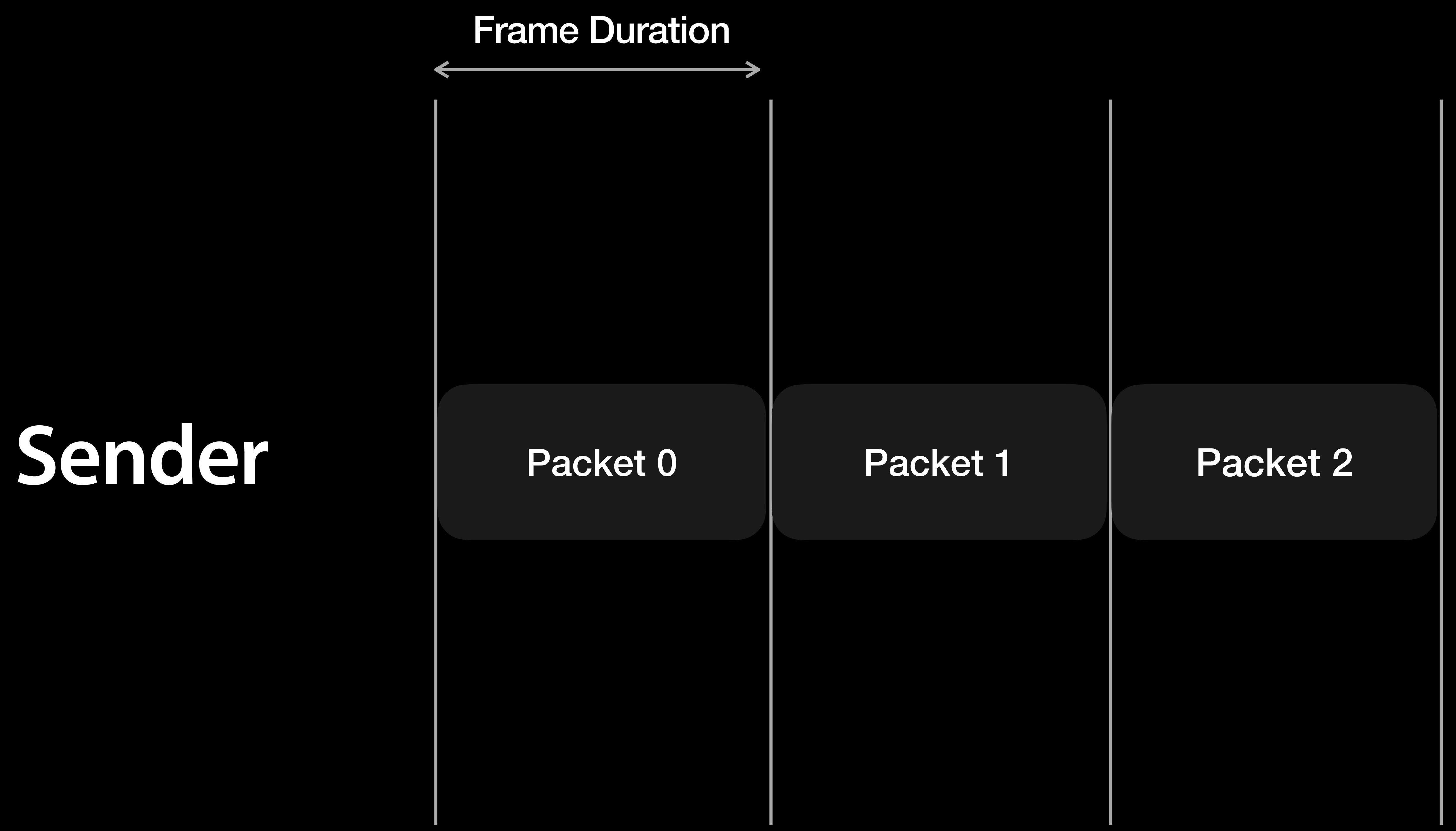
Receiver

System Architecture

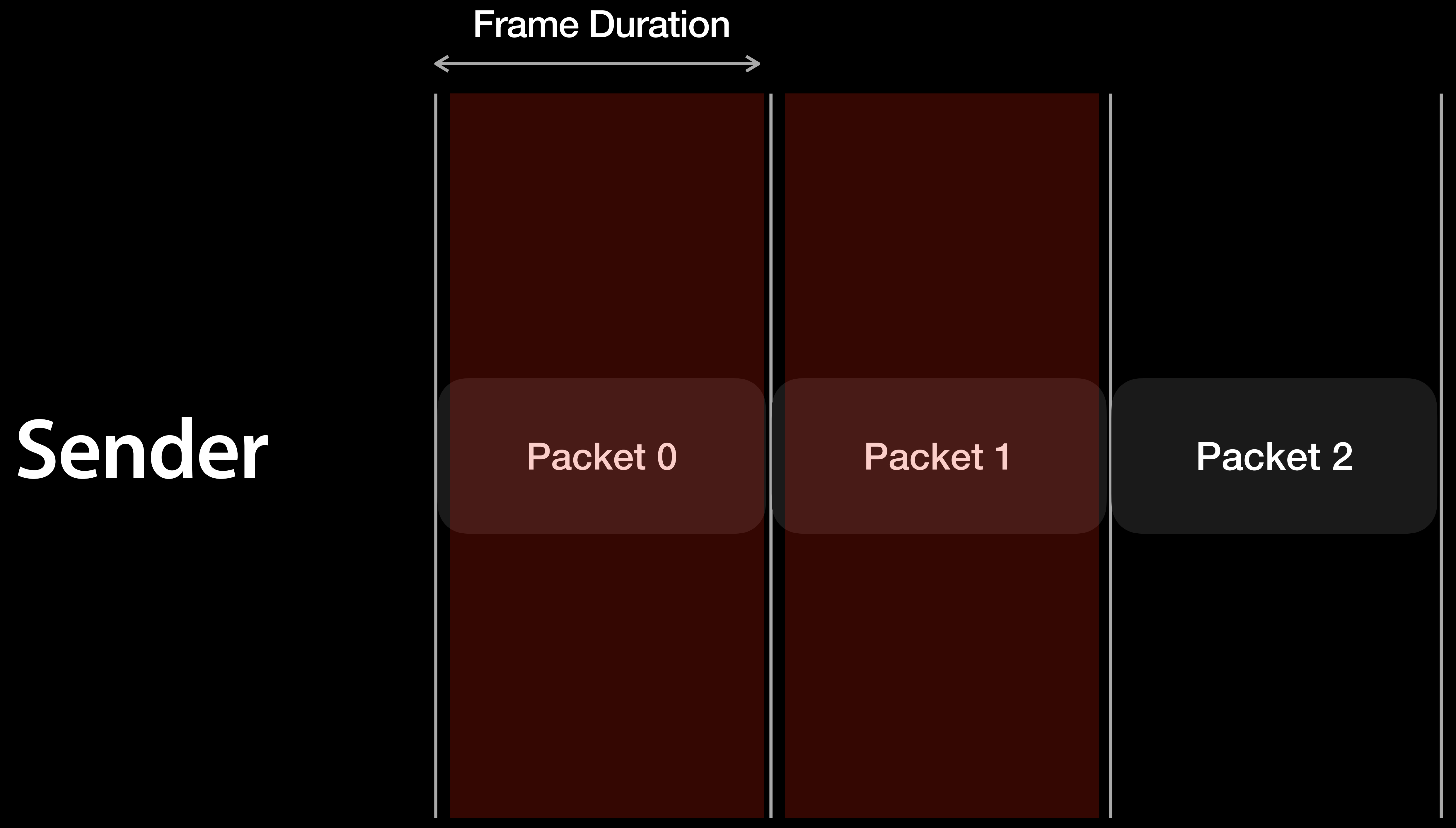


Video frame rate is usually twice the code display frame rate.

When video frame rate = code display frame rate

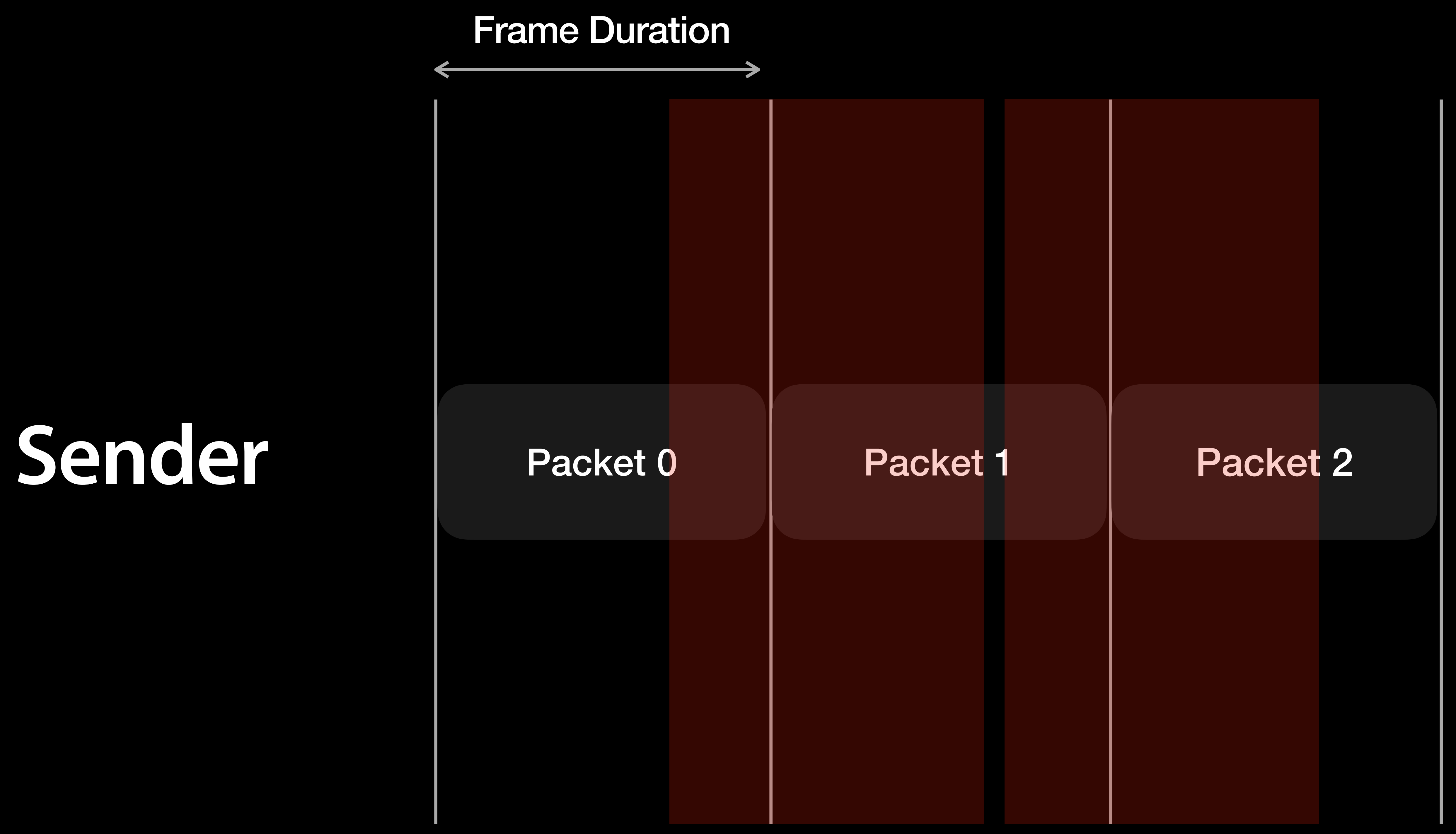


When video frame rate = code display frame rate



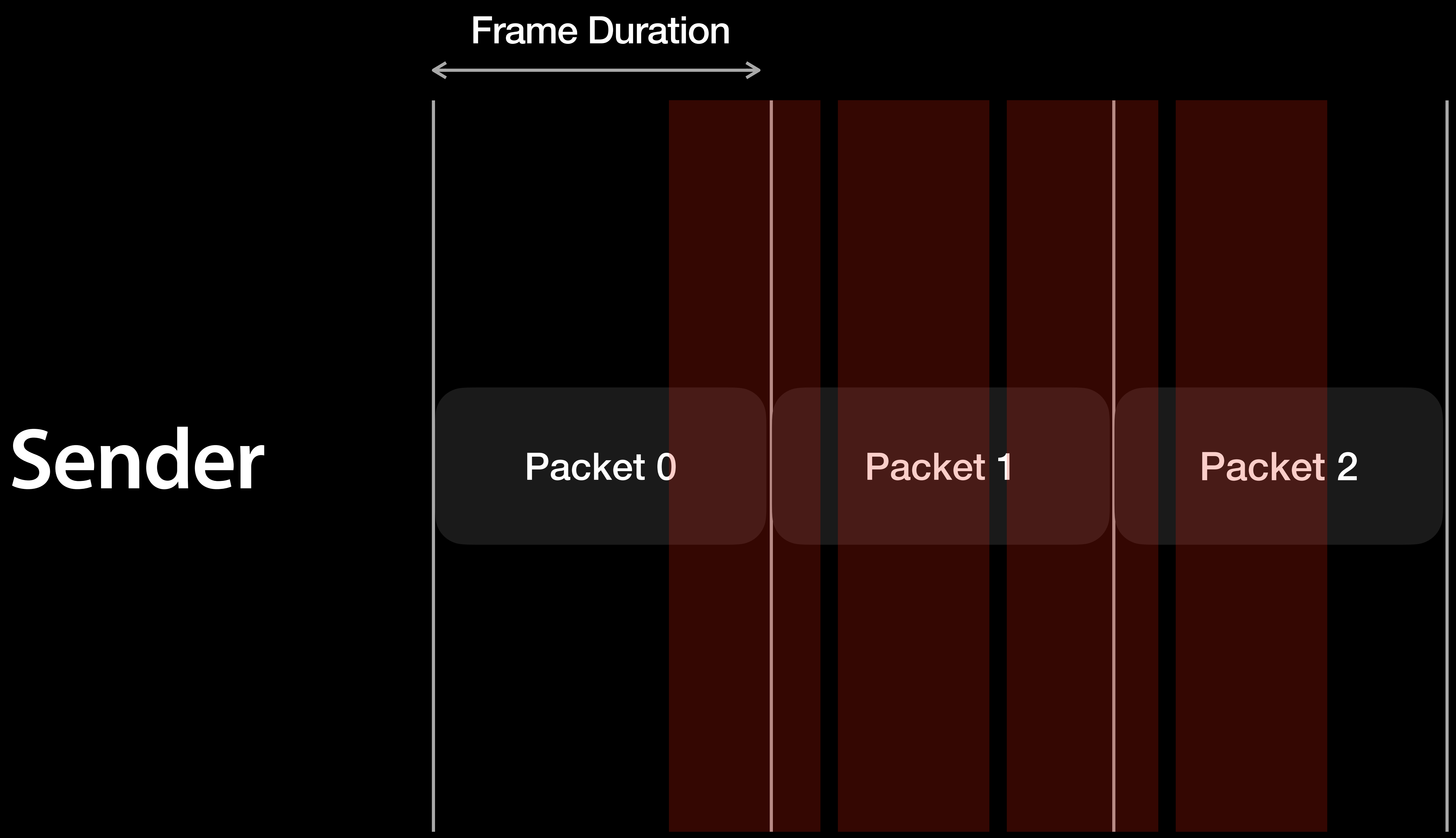
🟥 = **Camera Exposure Time**

When video frame rate = code display frame rate



🟥 = Camera Exposure Time

When video frame rate = 2 × code display frame rate



🟥 = Camera Exposure Time

When video frame rate = 2 × code display frame rate

Is it possible to prevent capturing mixed frames

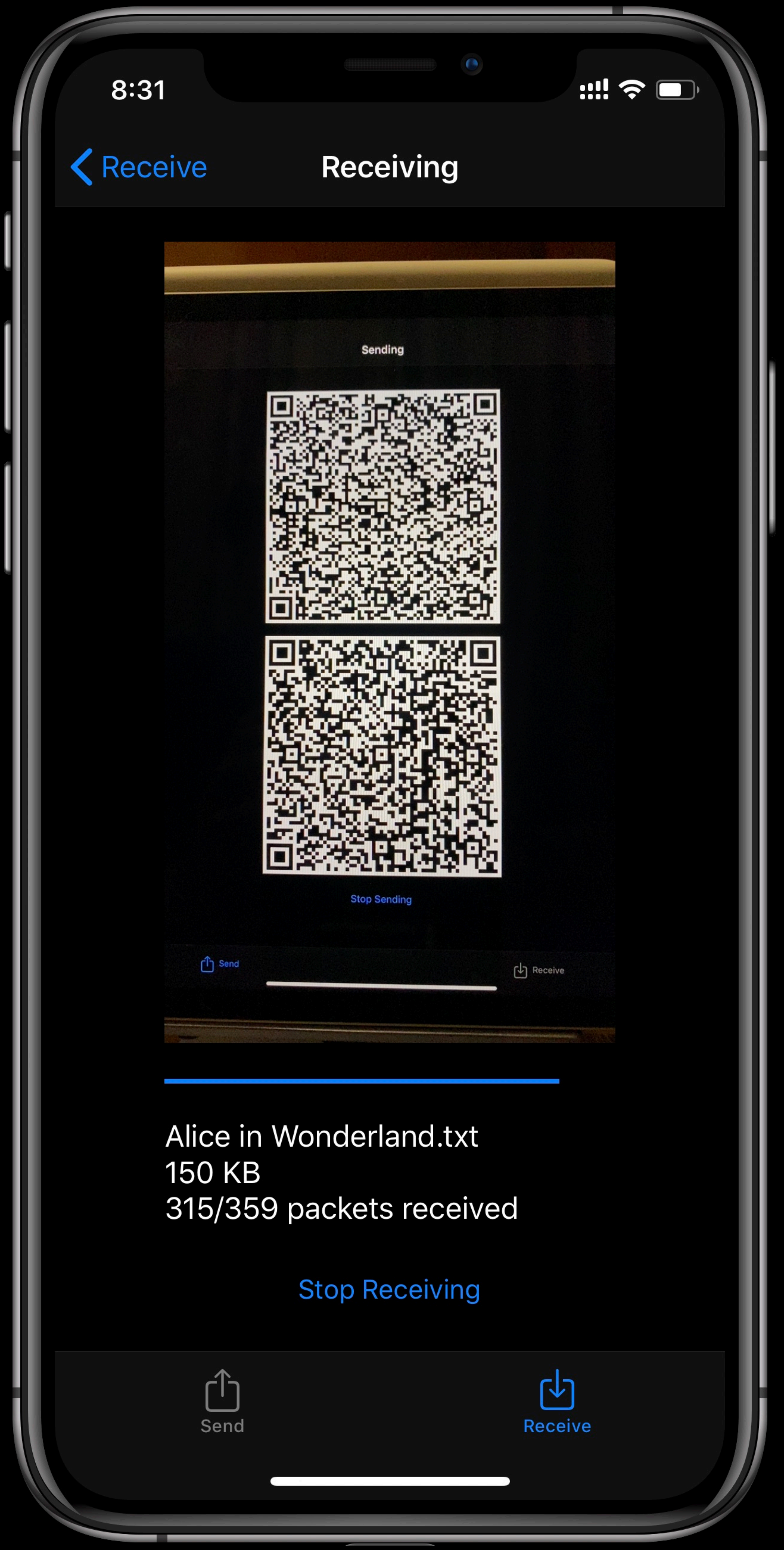
while keeping the video frame rate unchanged?

When video frame rate = 2 × code display frame rate

Is it possible to prevent capturing mixed frames while keeping the video frame rate unchanged?

Yes.

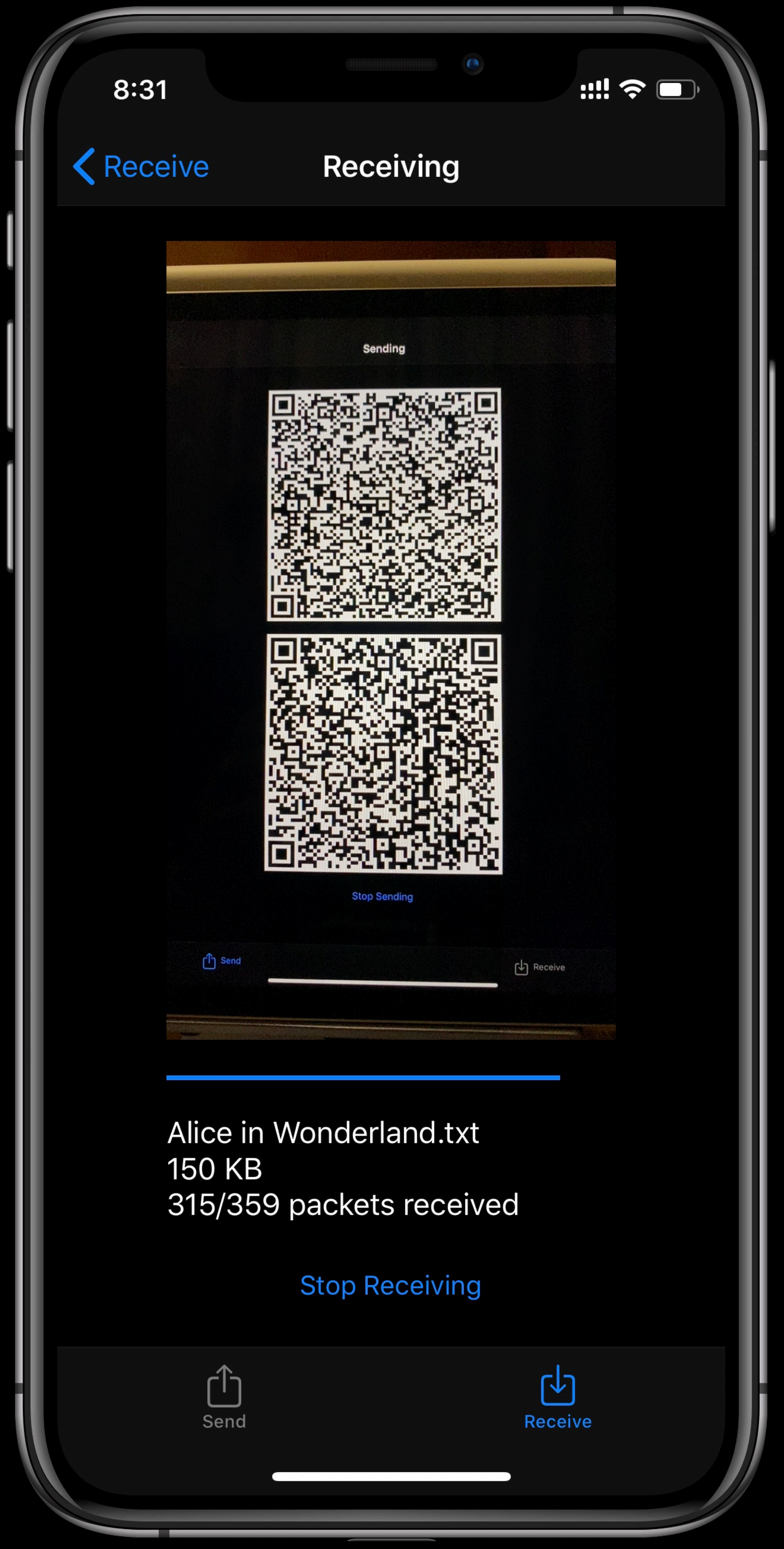
Alternating Code Display Mode



Alice in Wonderland.txt
150 KB
315/359 packets received

Alternating Code Display Mode

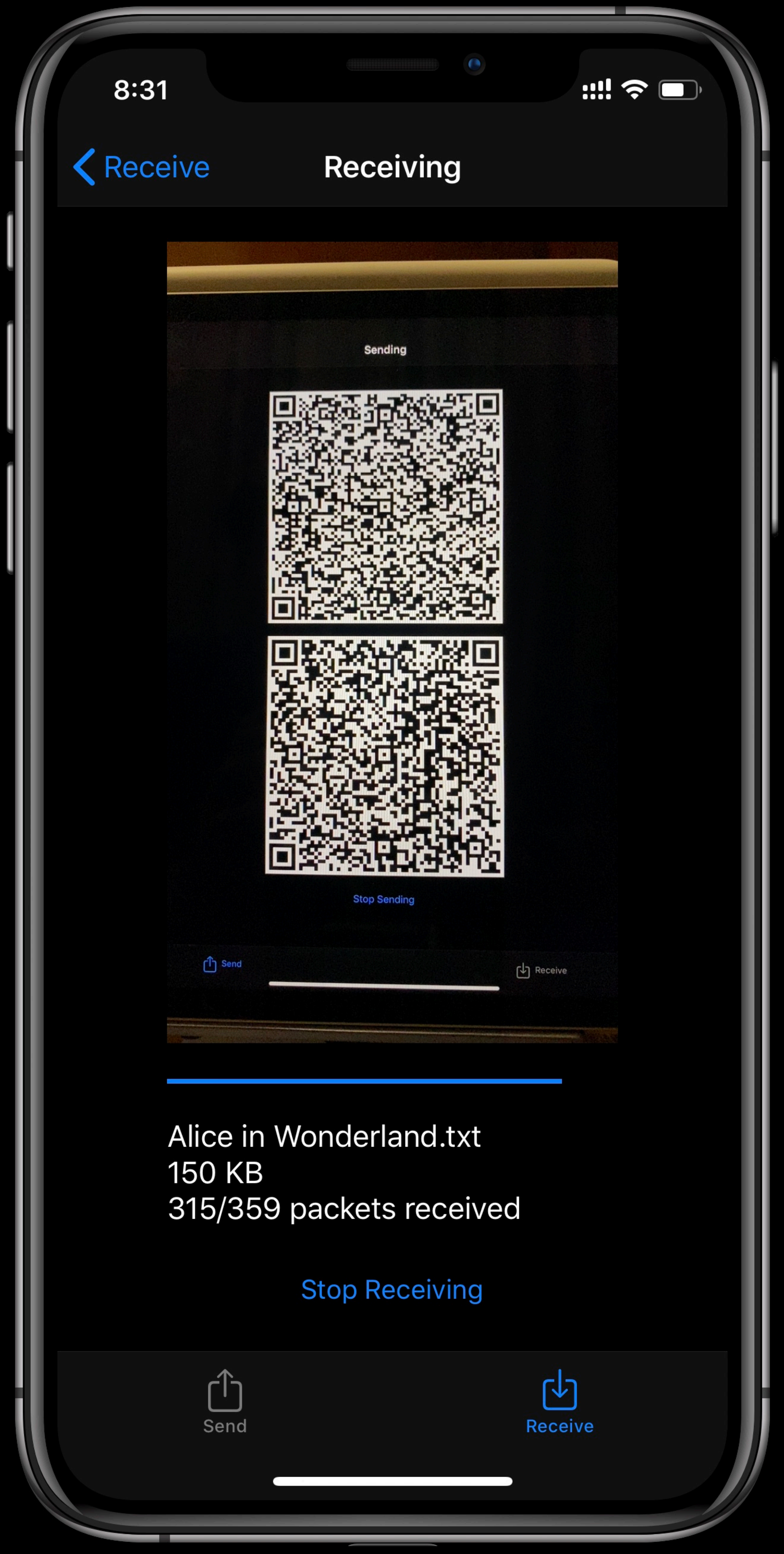
- Two views for displaying codes



Alice in Wonderland.txt
150 KB
315/359 packets received

Alternating Code Display Mode

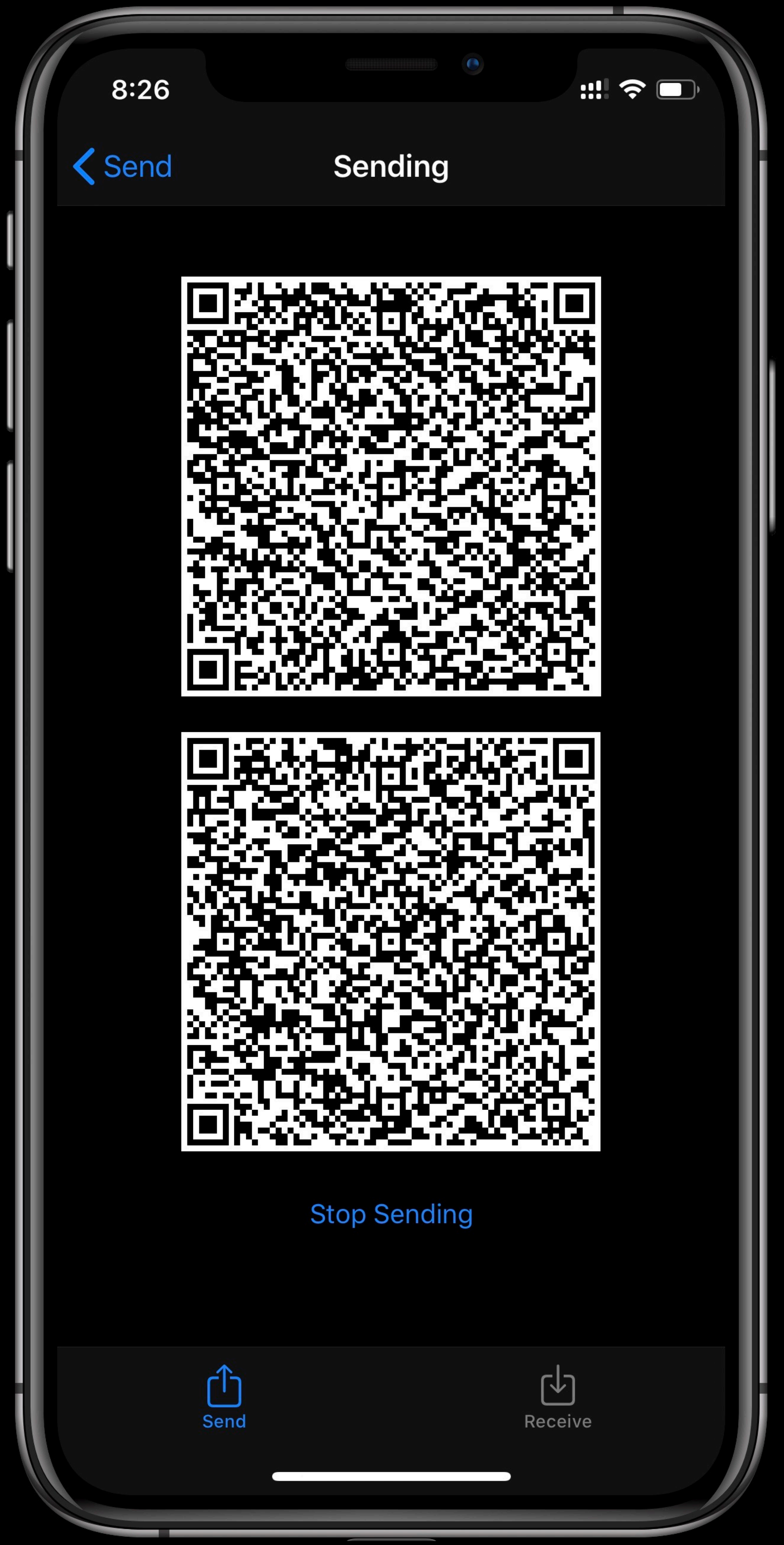
- Two views for displaying codes
- Used in an alternating manner



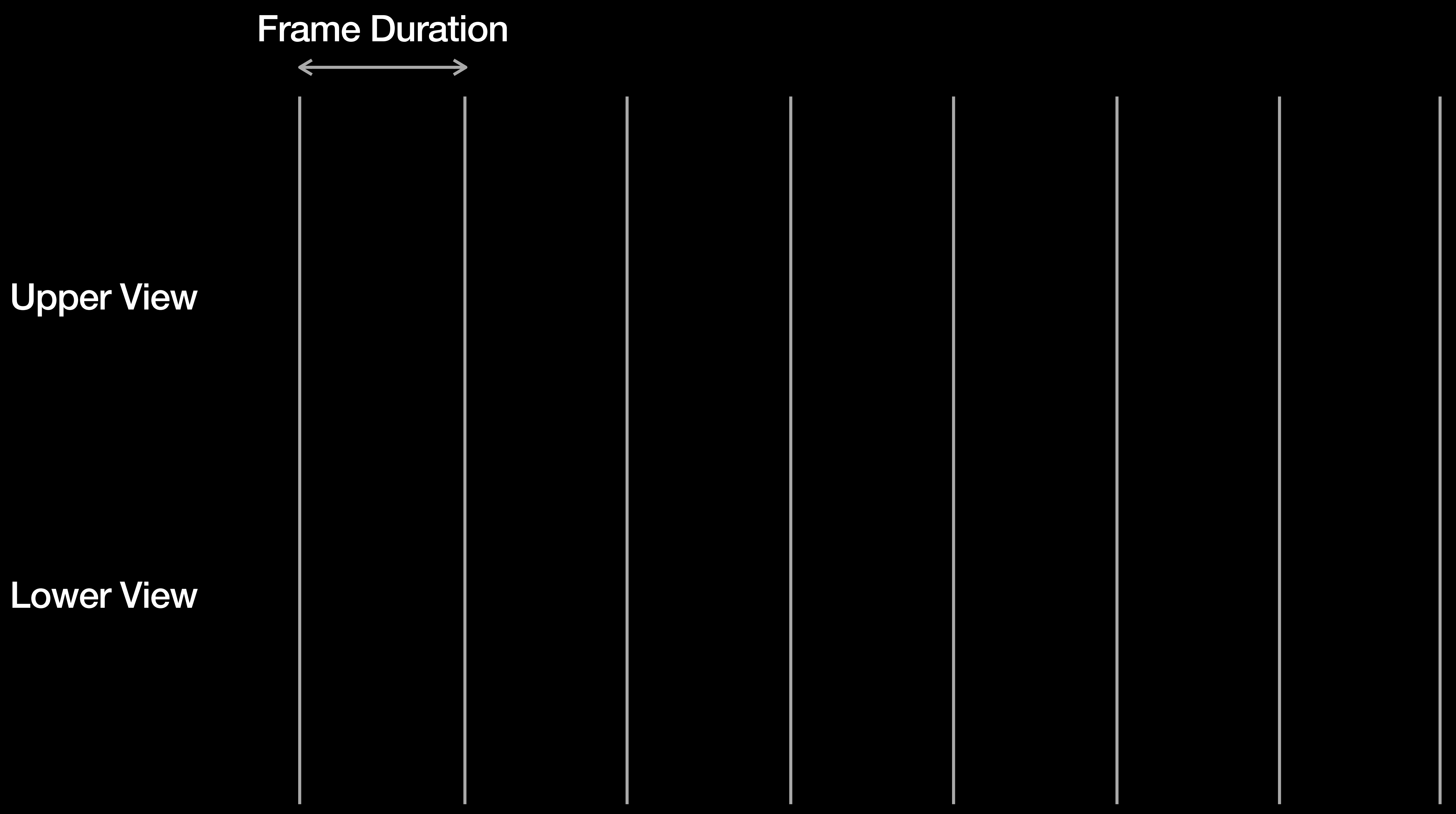
Alice in Wonderland.txt
150 KB
315/359 packets received

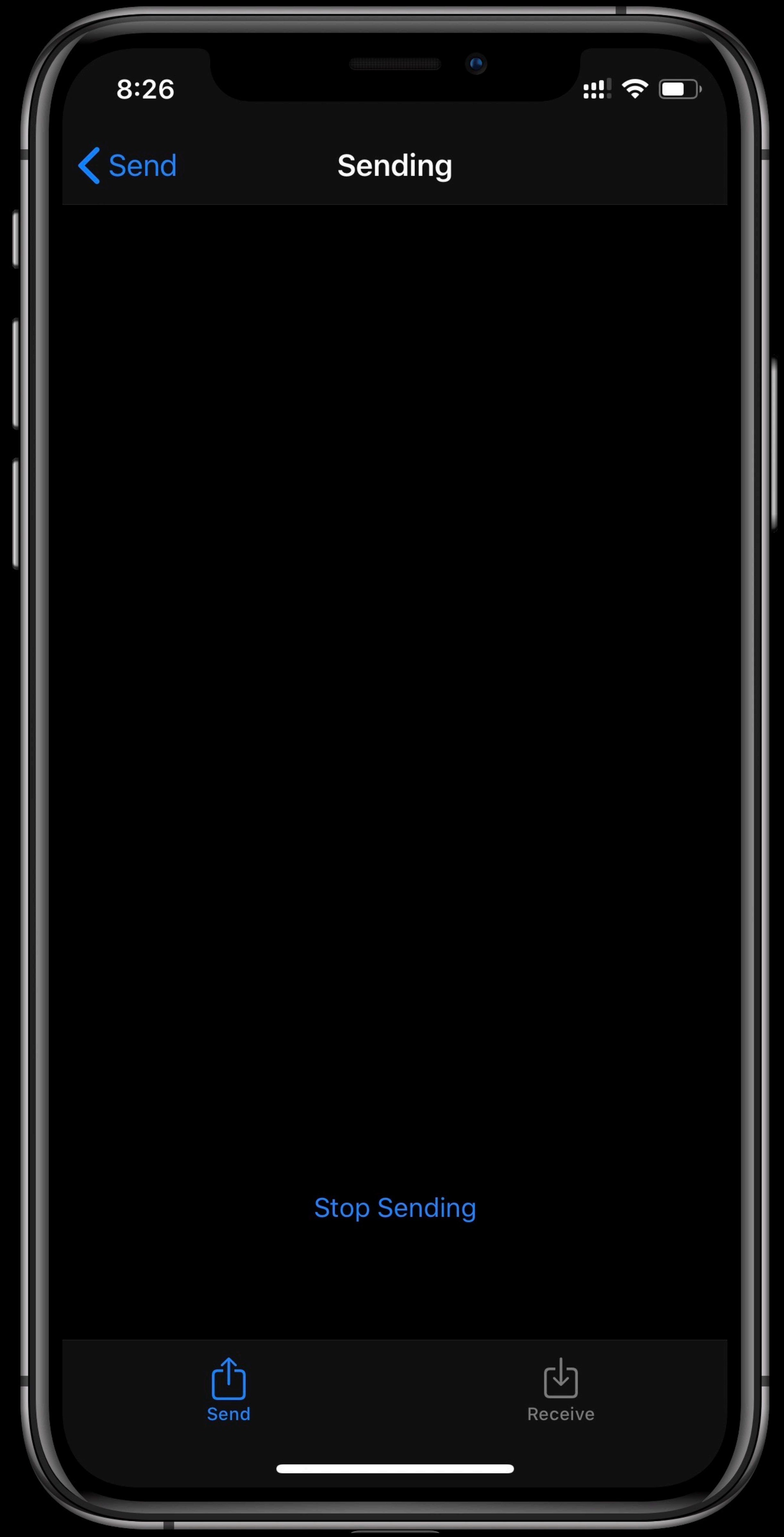
Alternating Code Display Mode

- Two views for displaying codes
- Used in an alternating manner

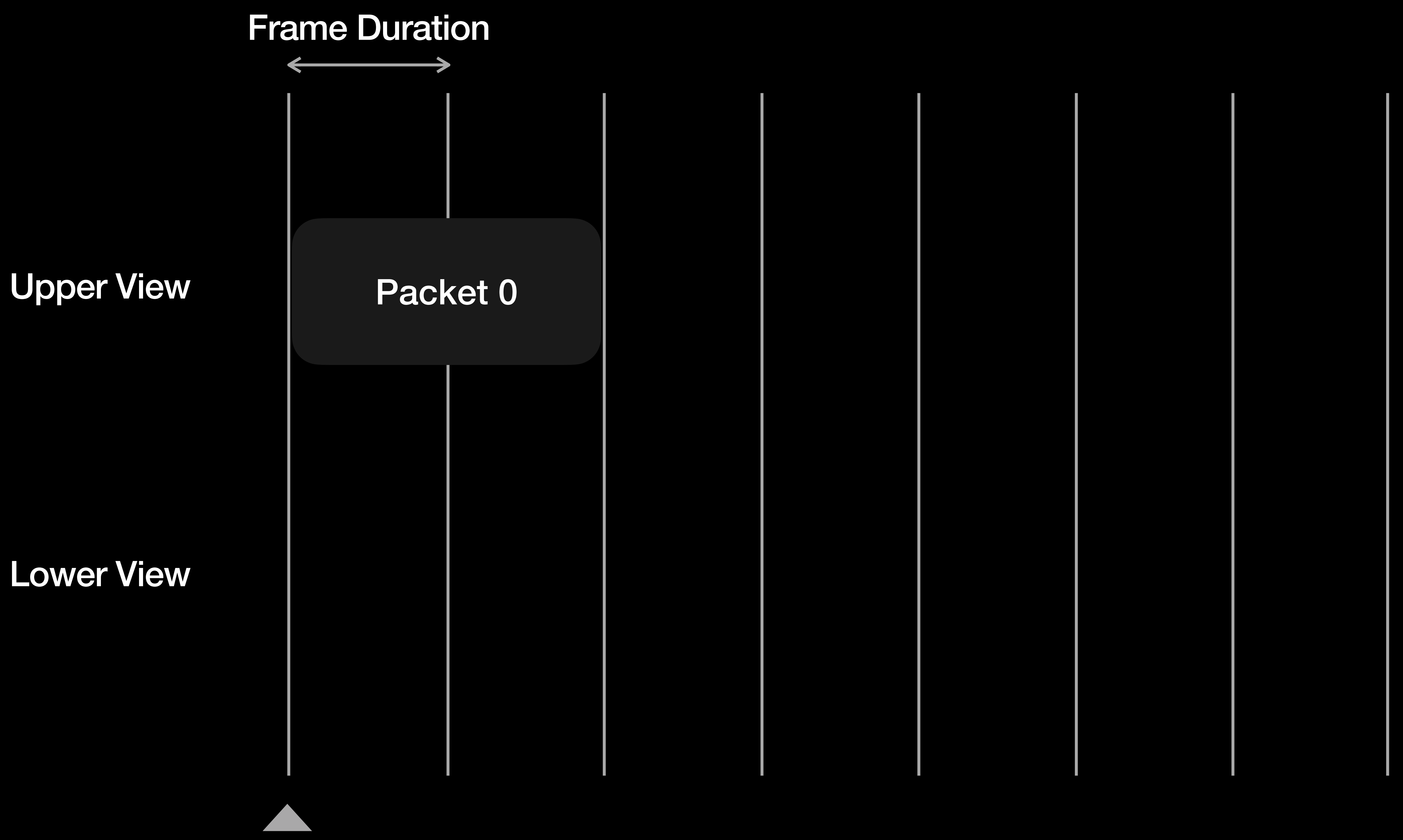


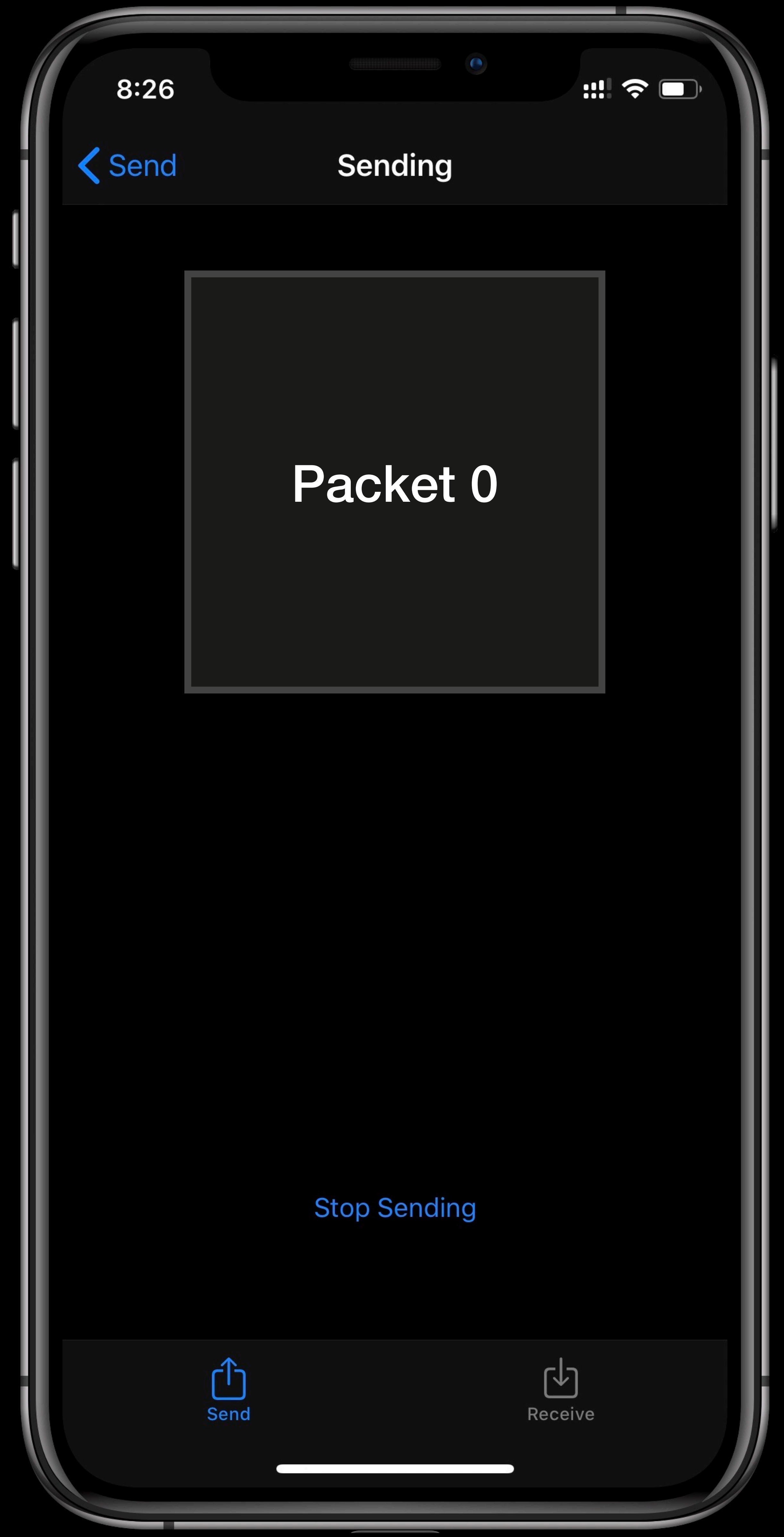
Alternating Code Display Mode



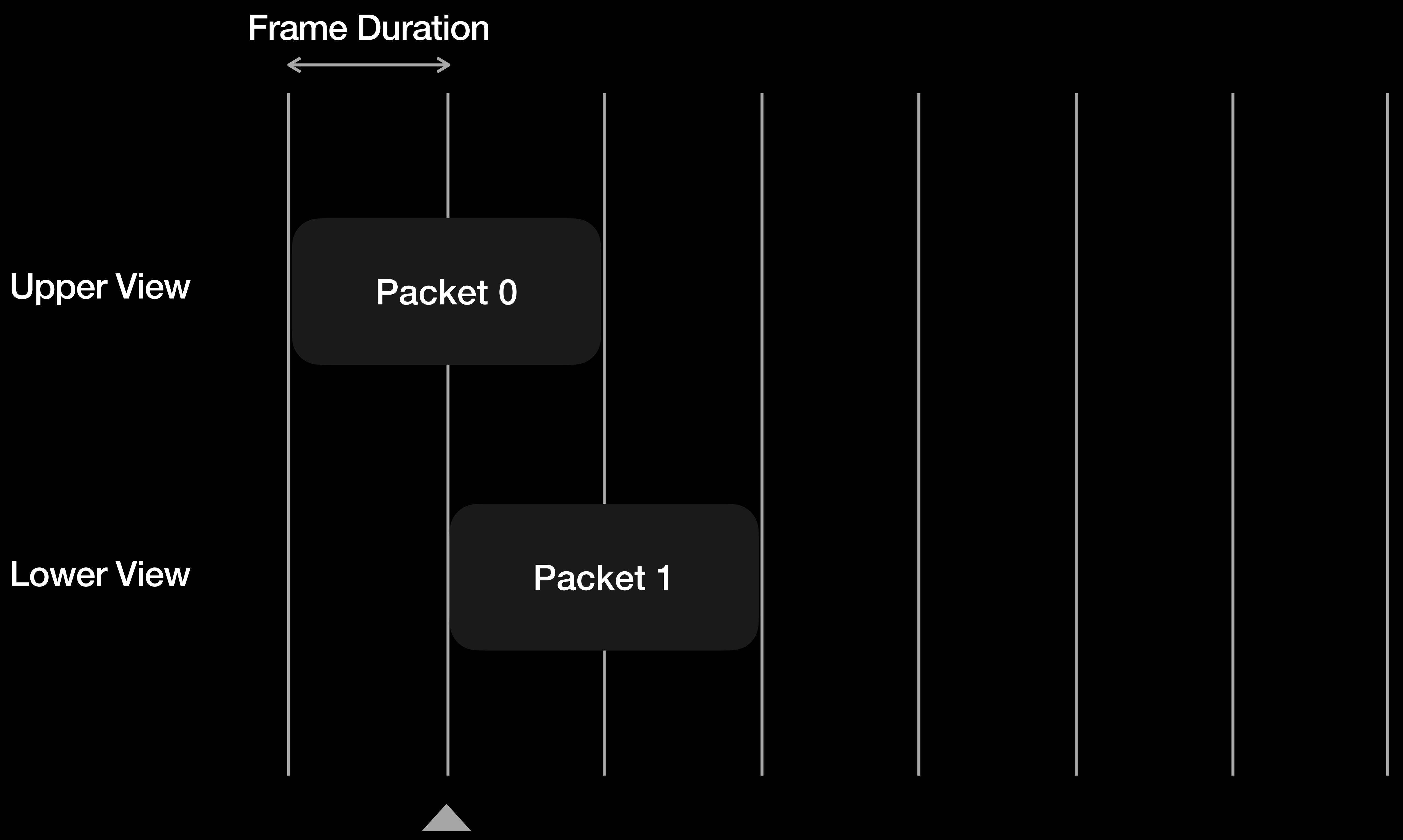


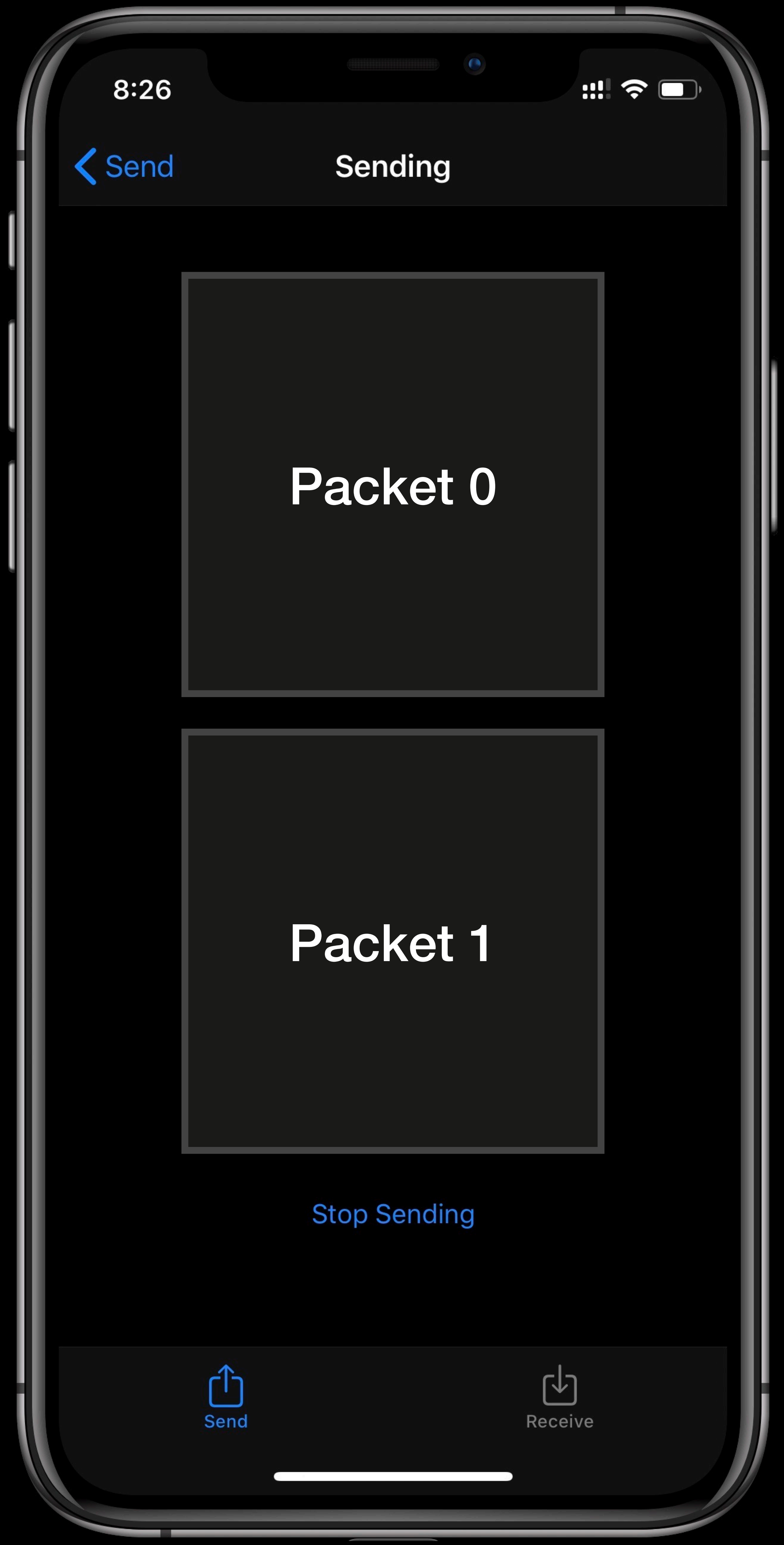
Alternating Code Display Mode



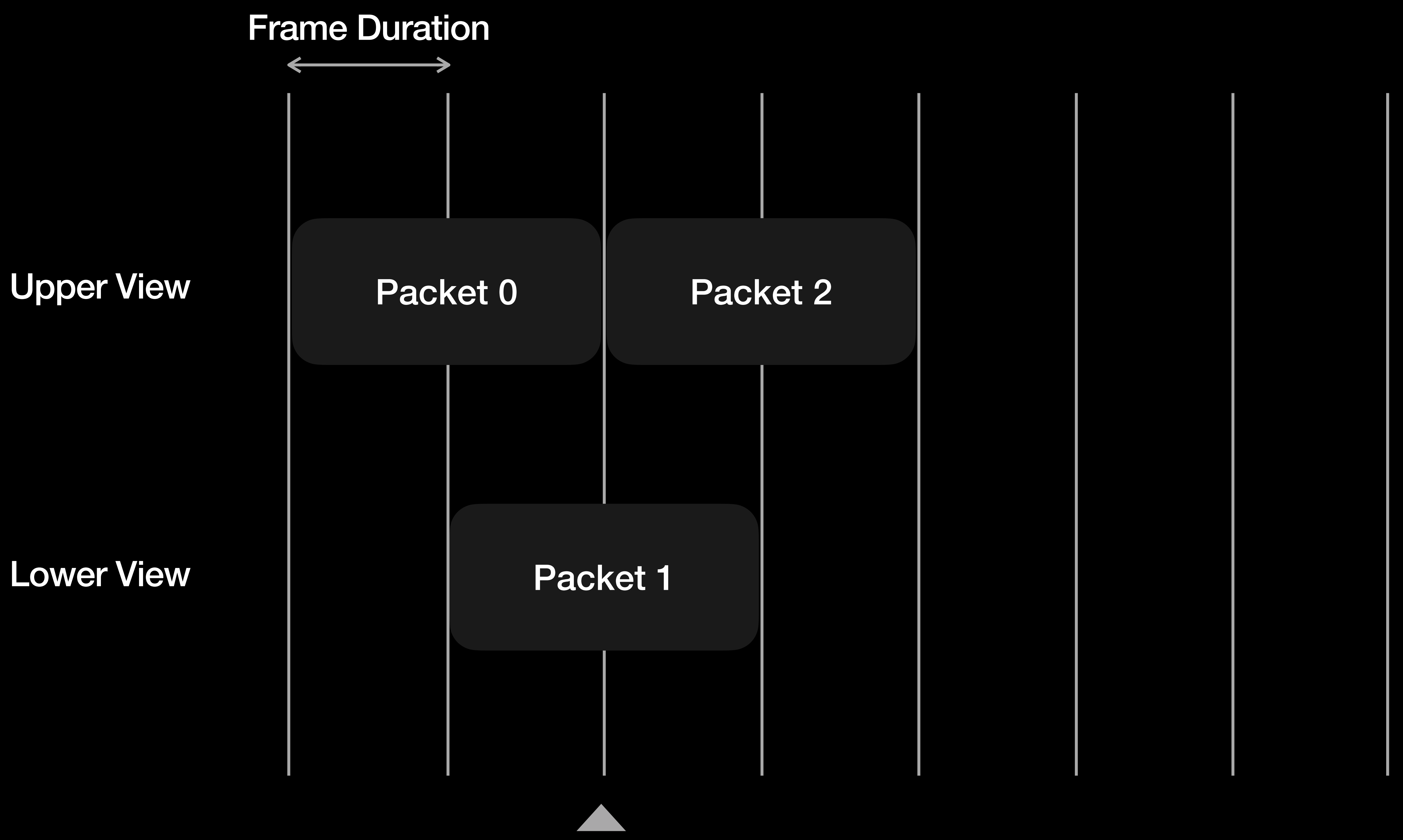


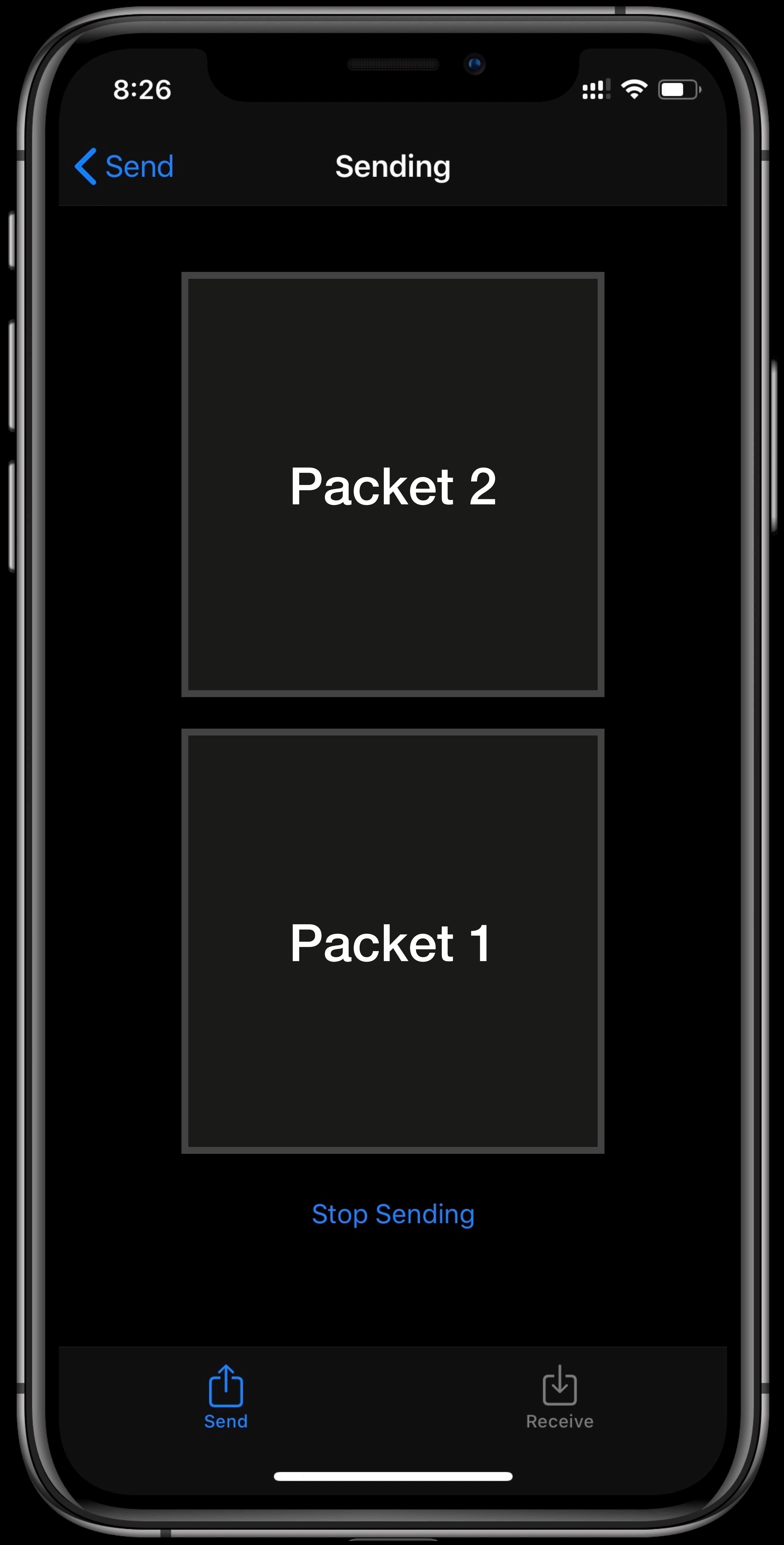
Alternating Code Display Mode



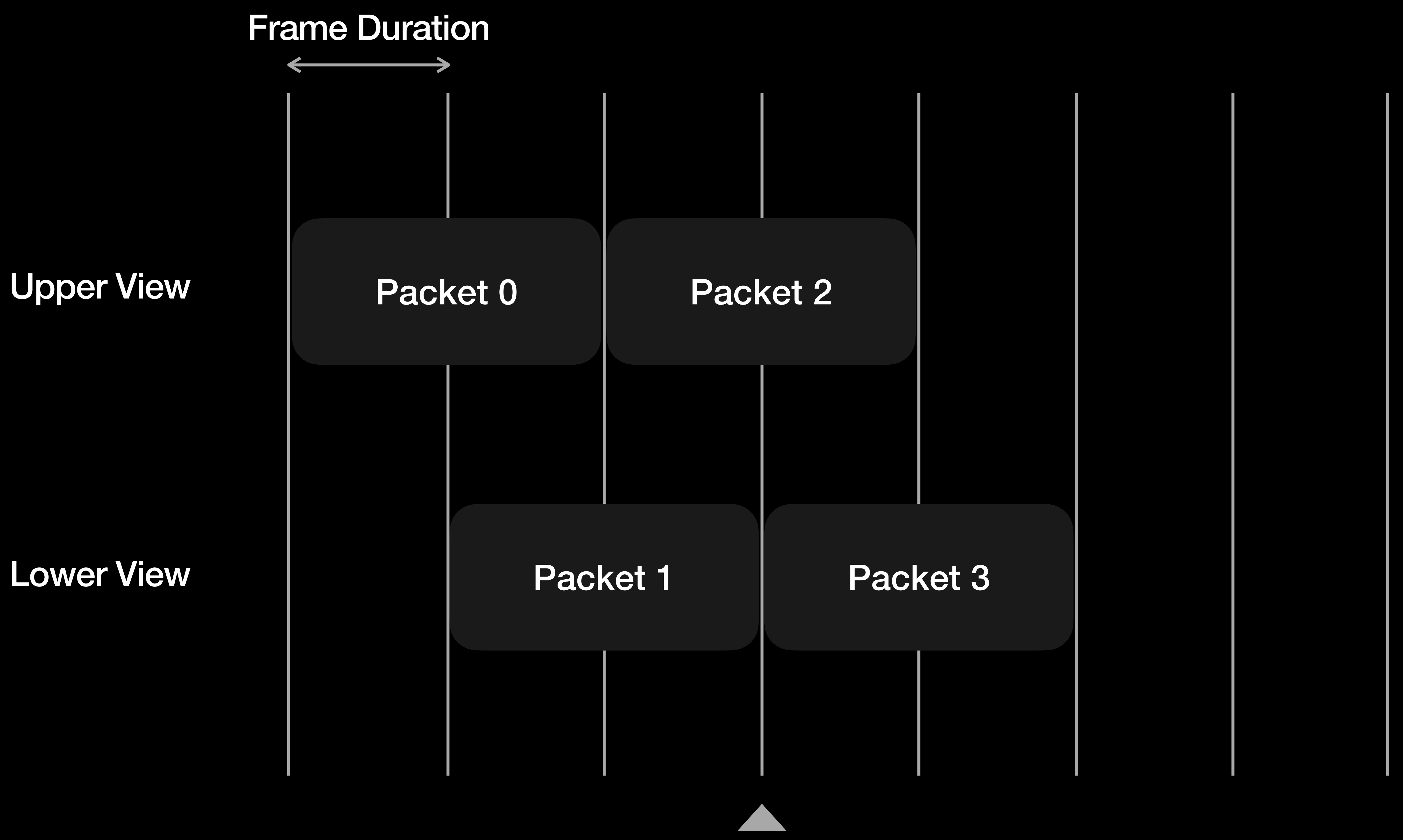


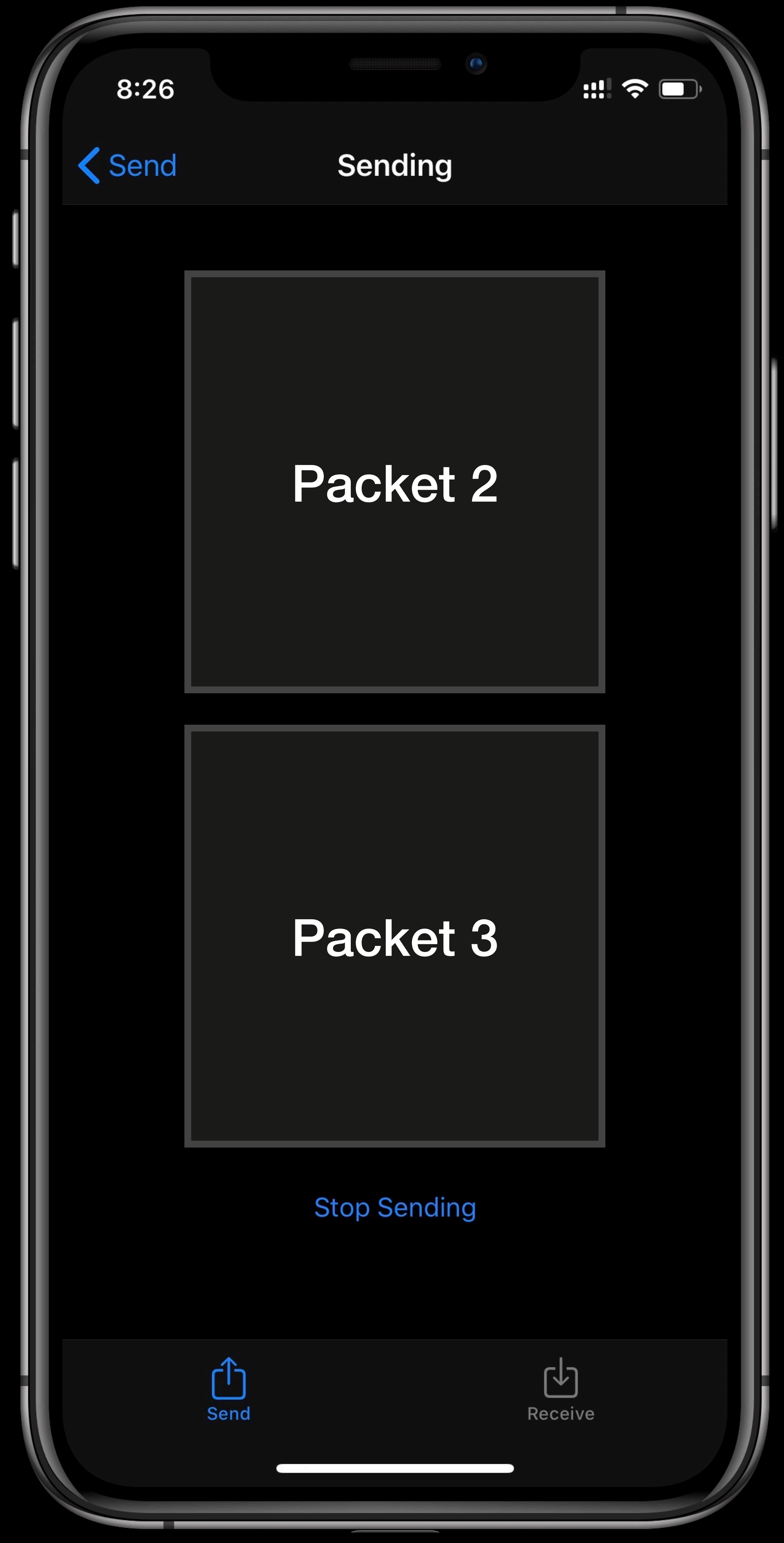
Alternating Code Display Mode



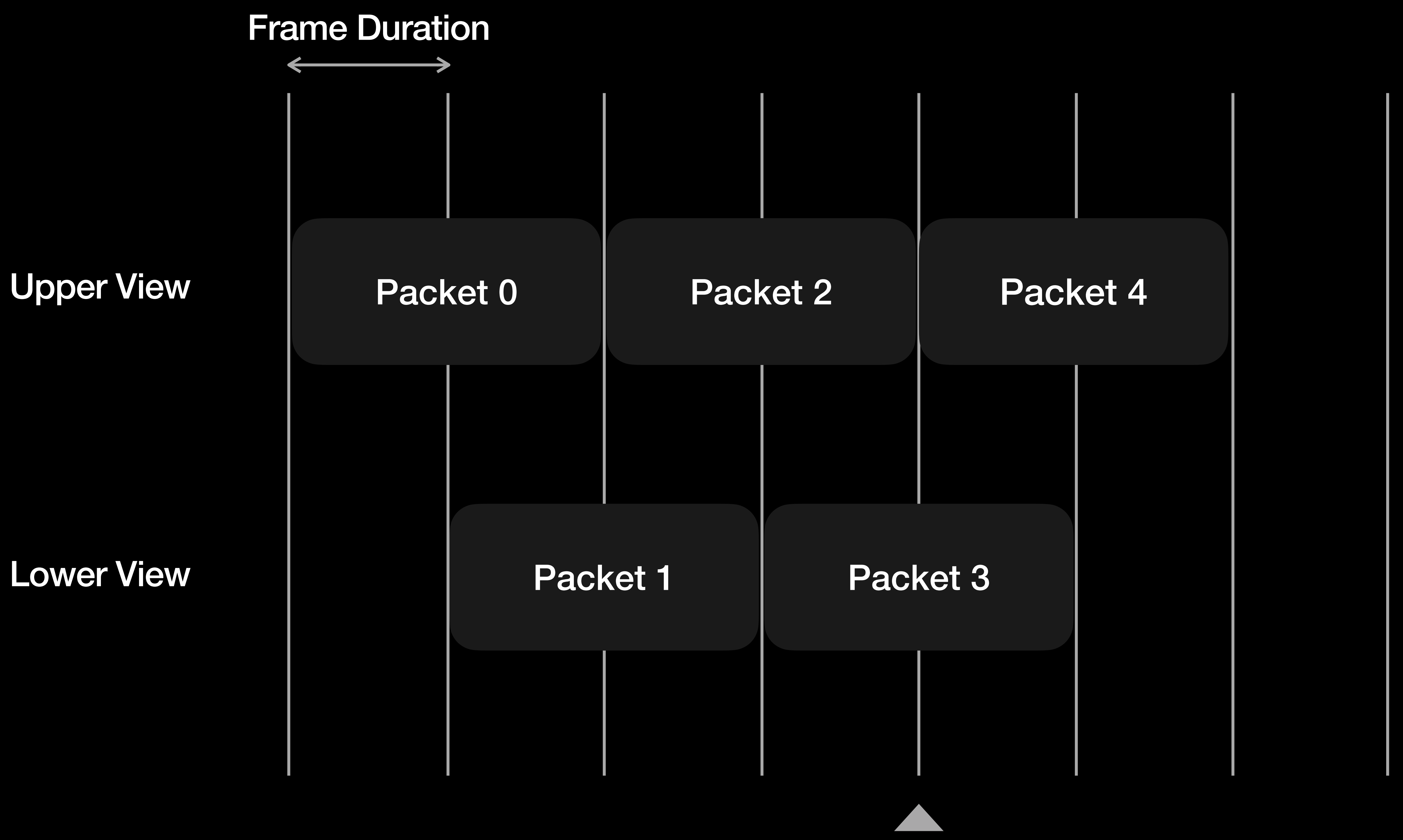


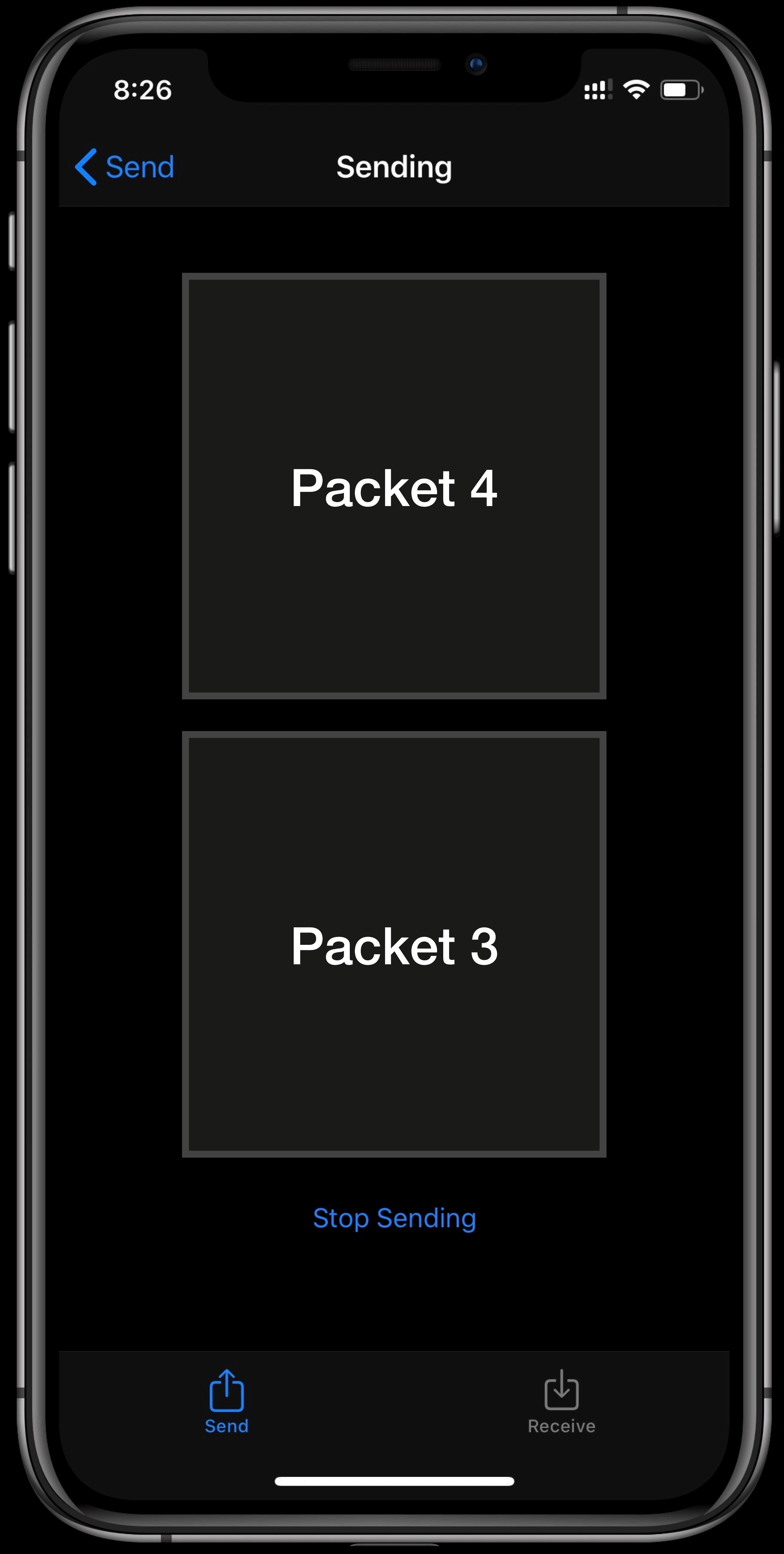
Alternating Code Display Mode



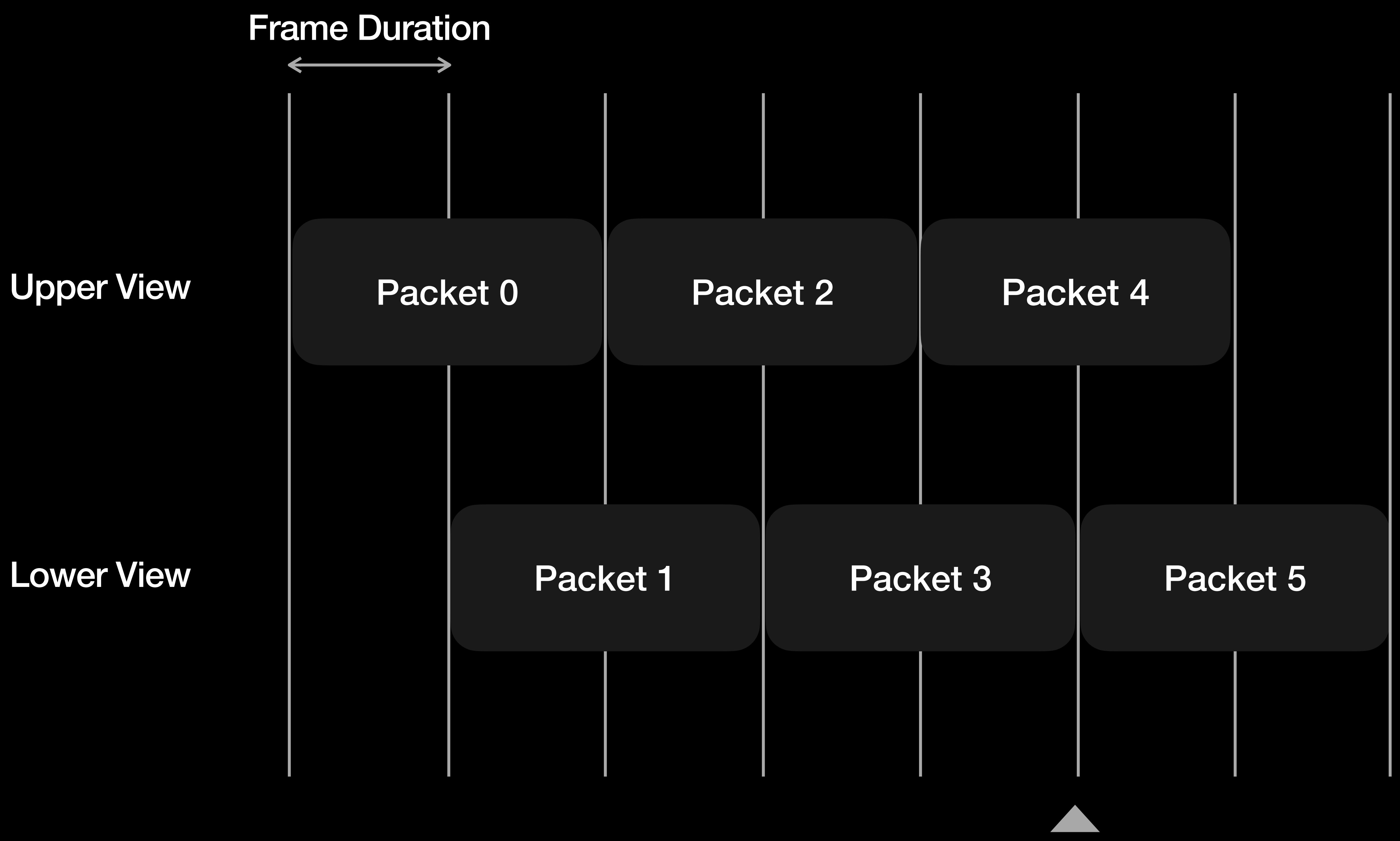


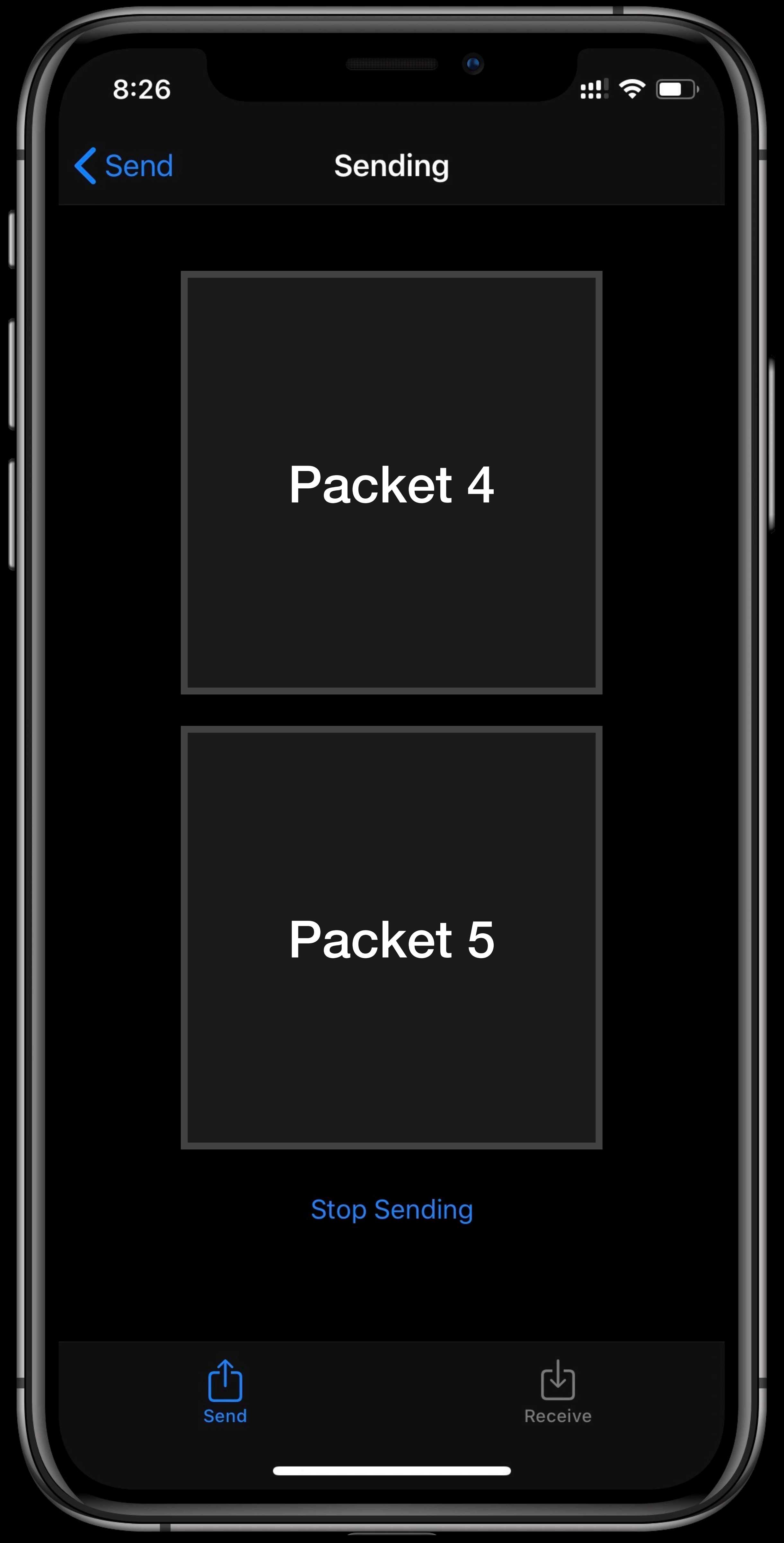
Alternating Code Display Mode



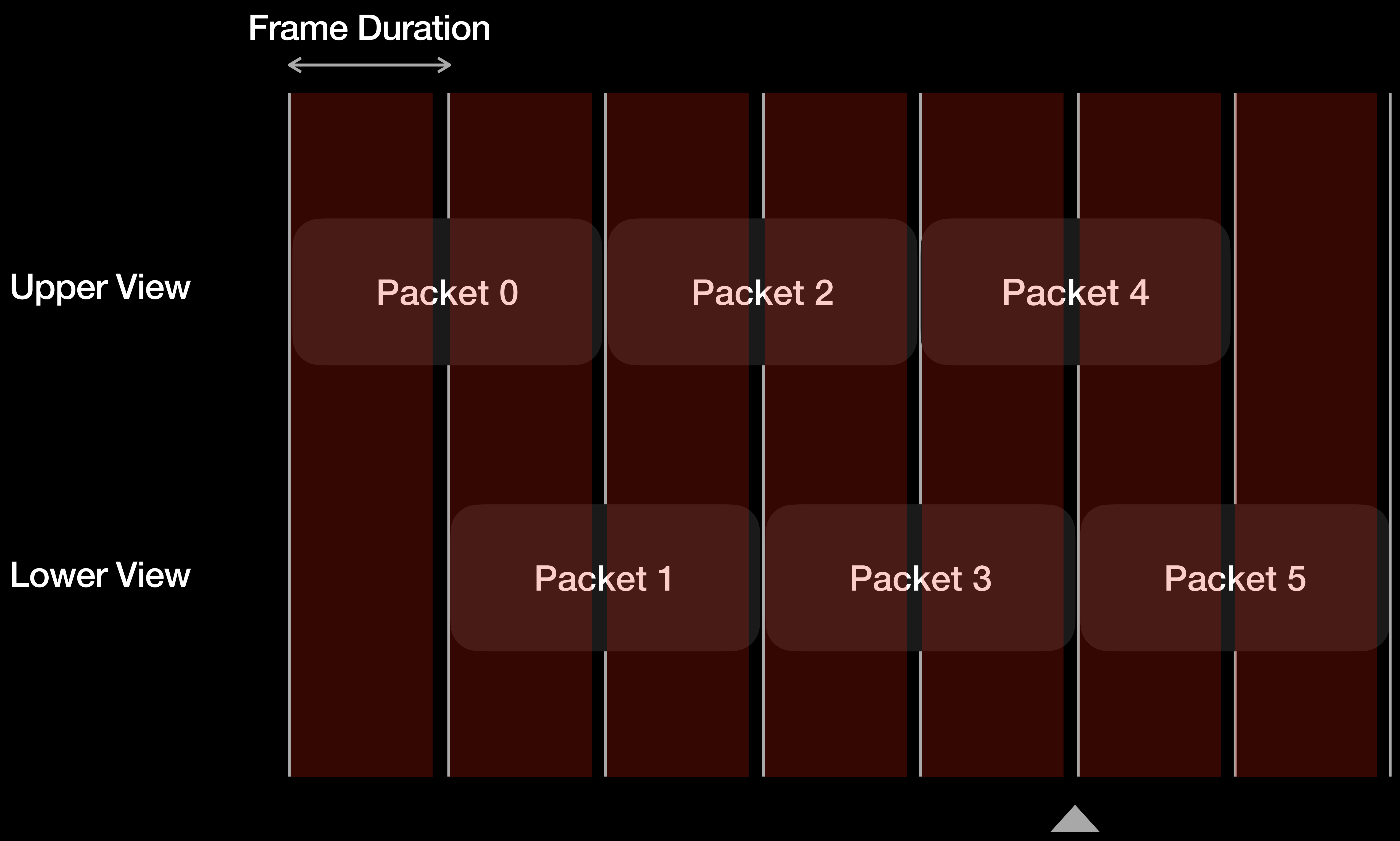


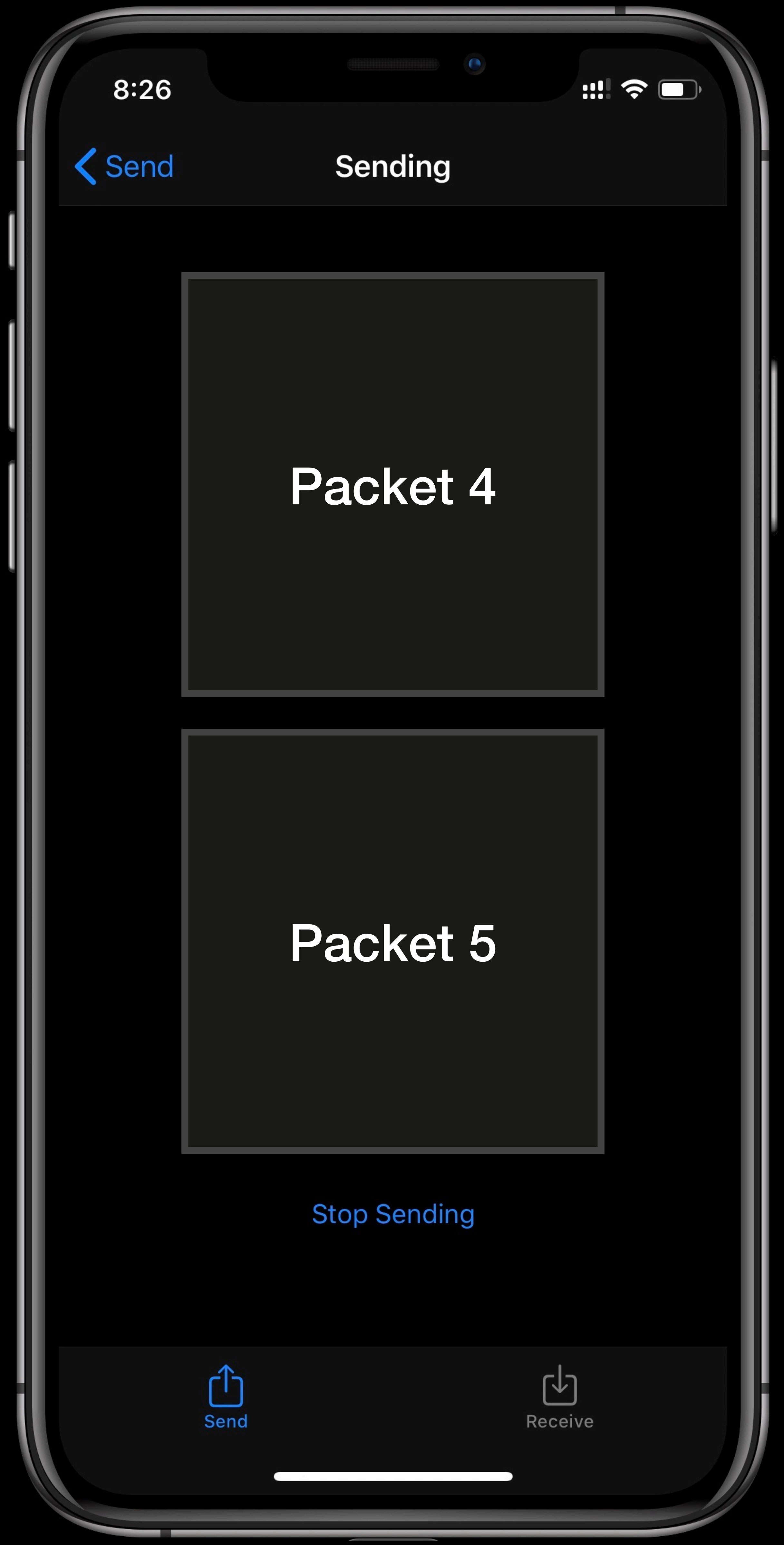
Alternating Code Display Mode



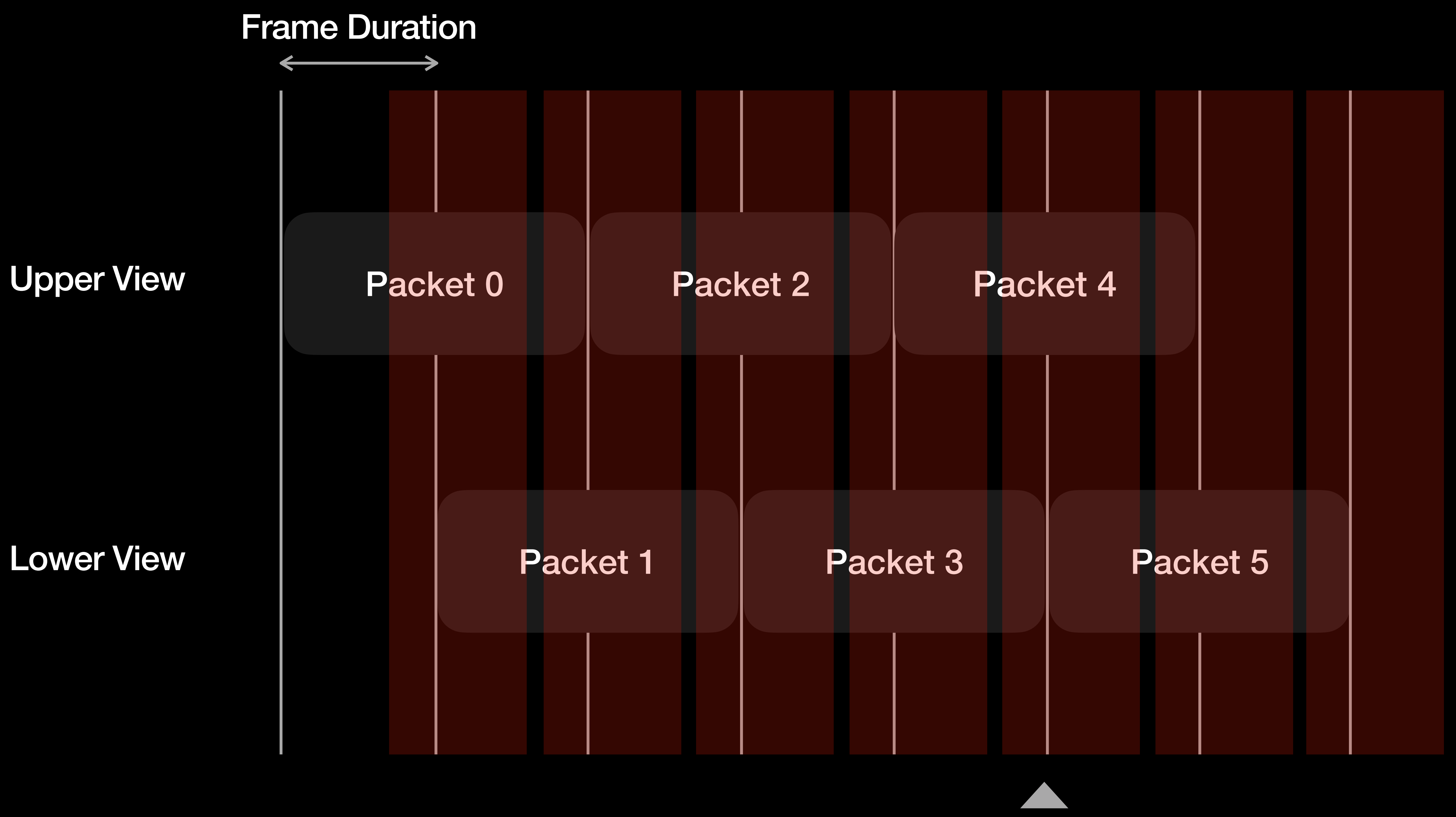


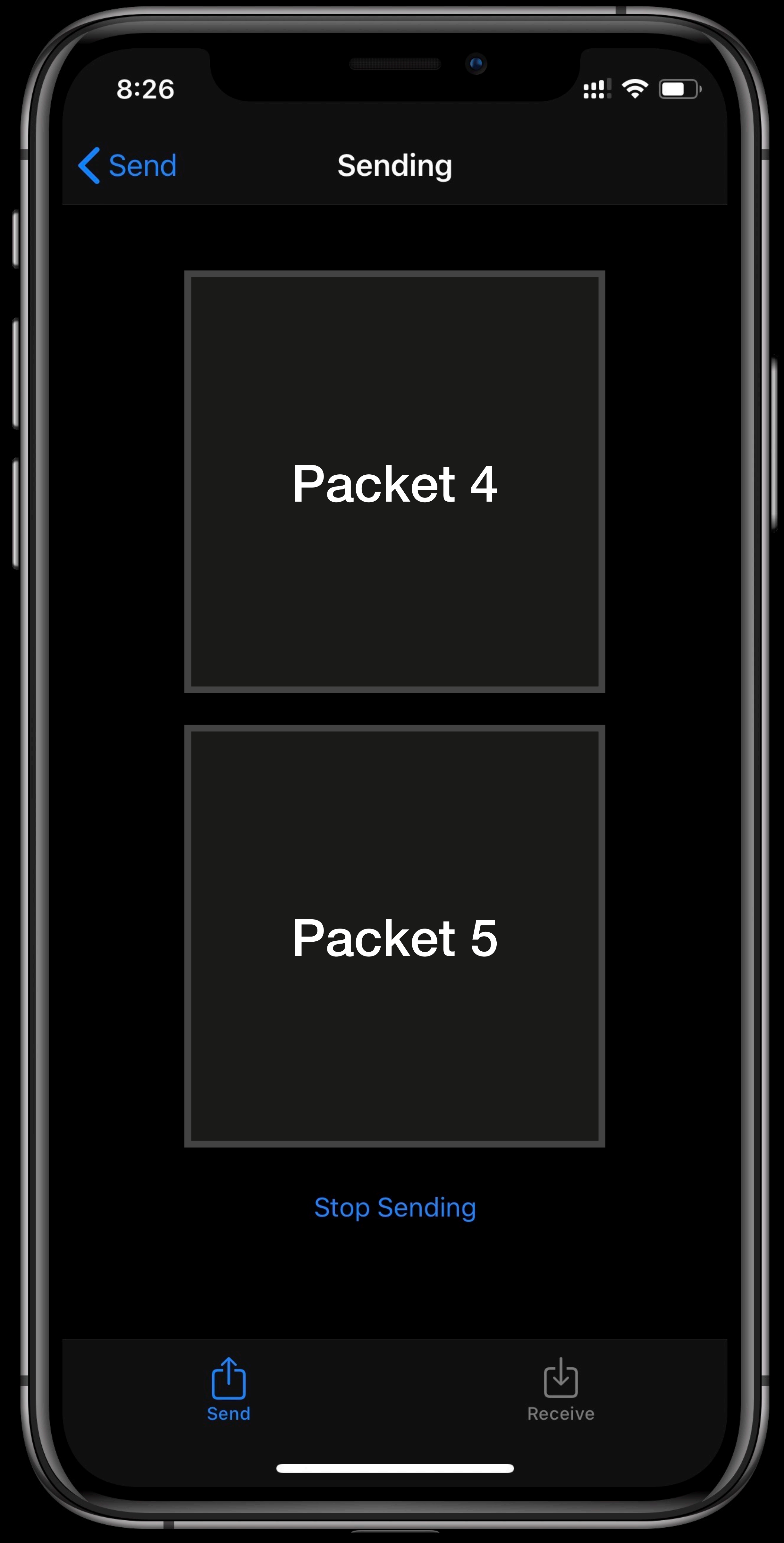
Alternating Code Display Mode





Alternating Code Display Mode

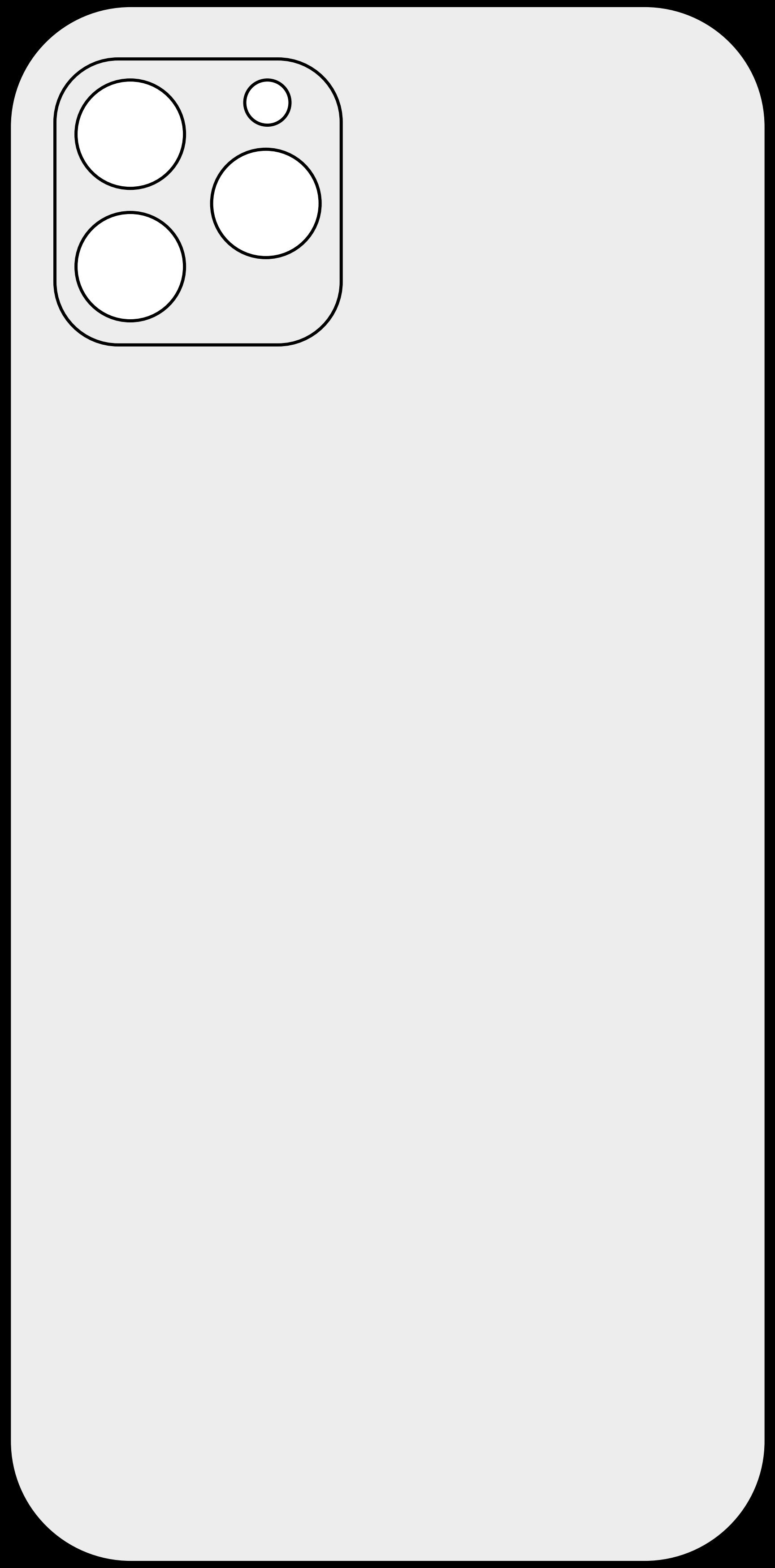




Nested Code Display Mode

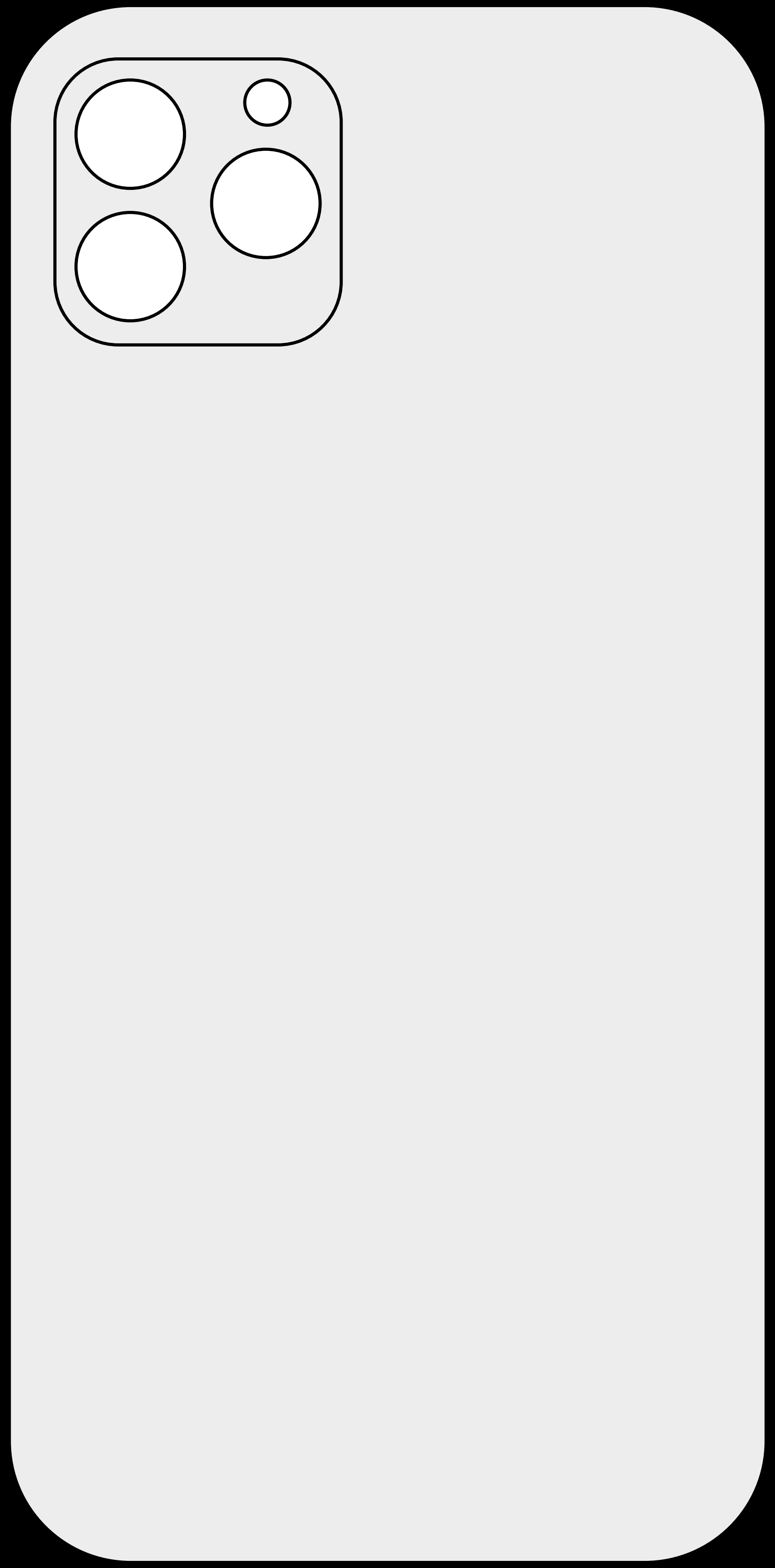
Nested Code Display Mode

- Recent smartphones are equipped with multiple cameras

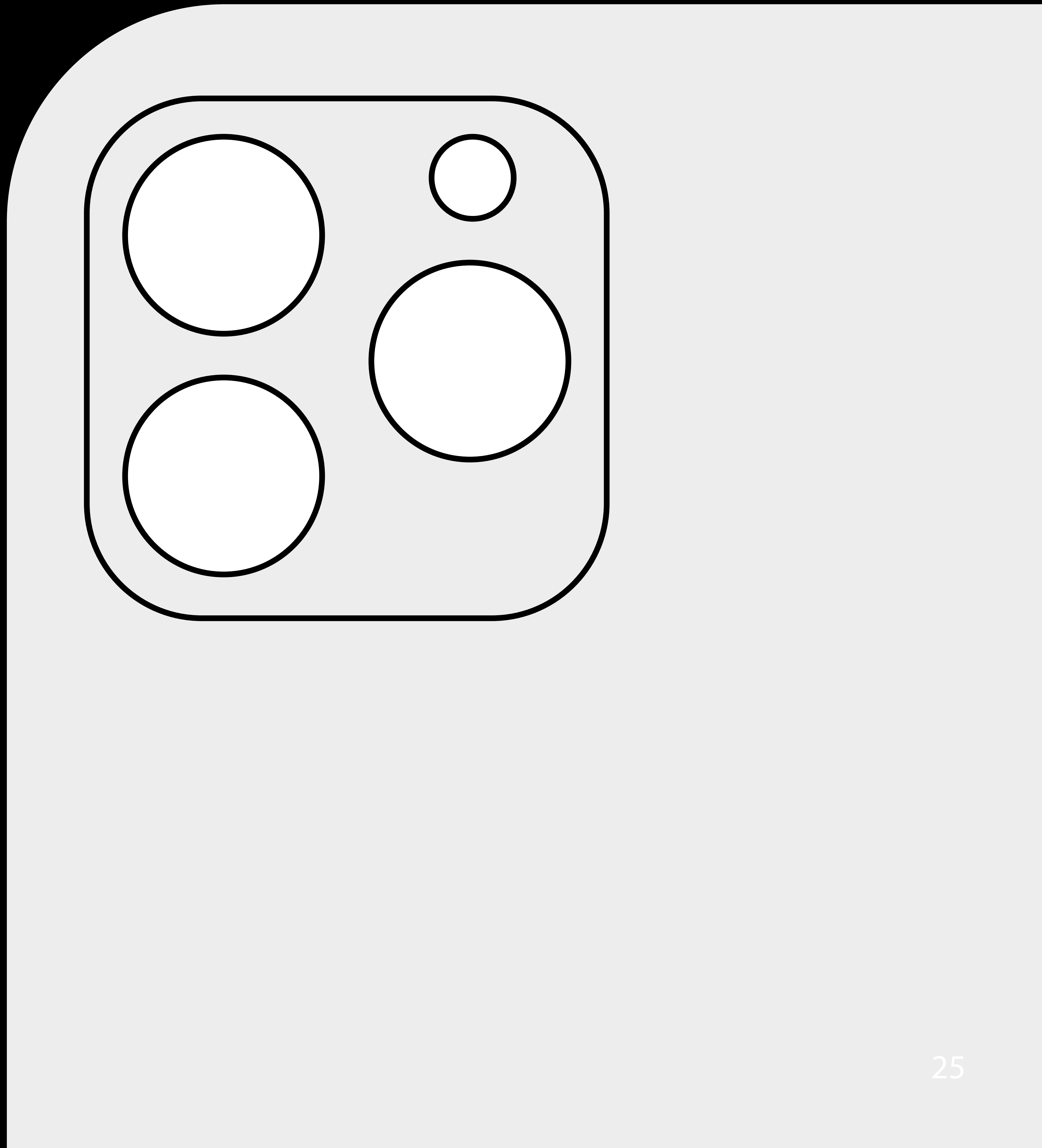


Nested Code Display Mode

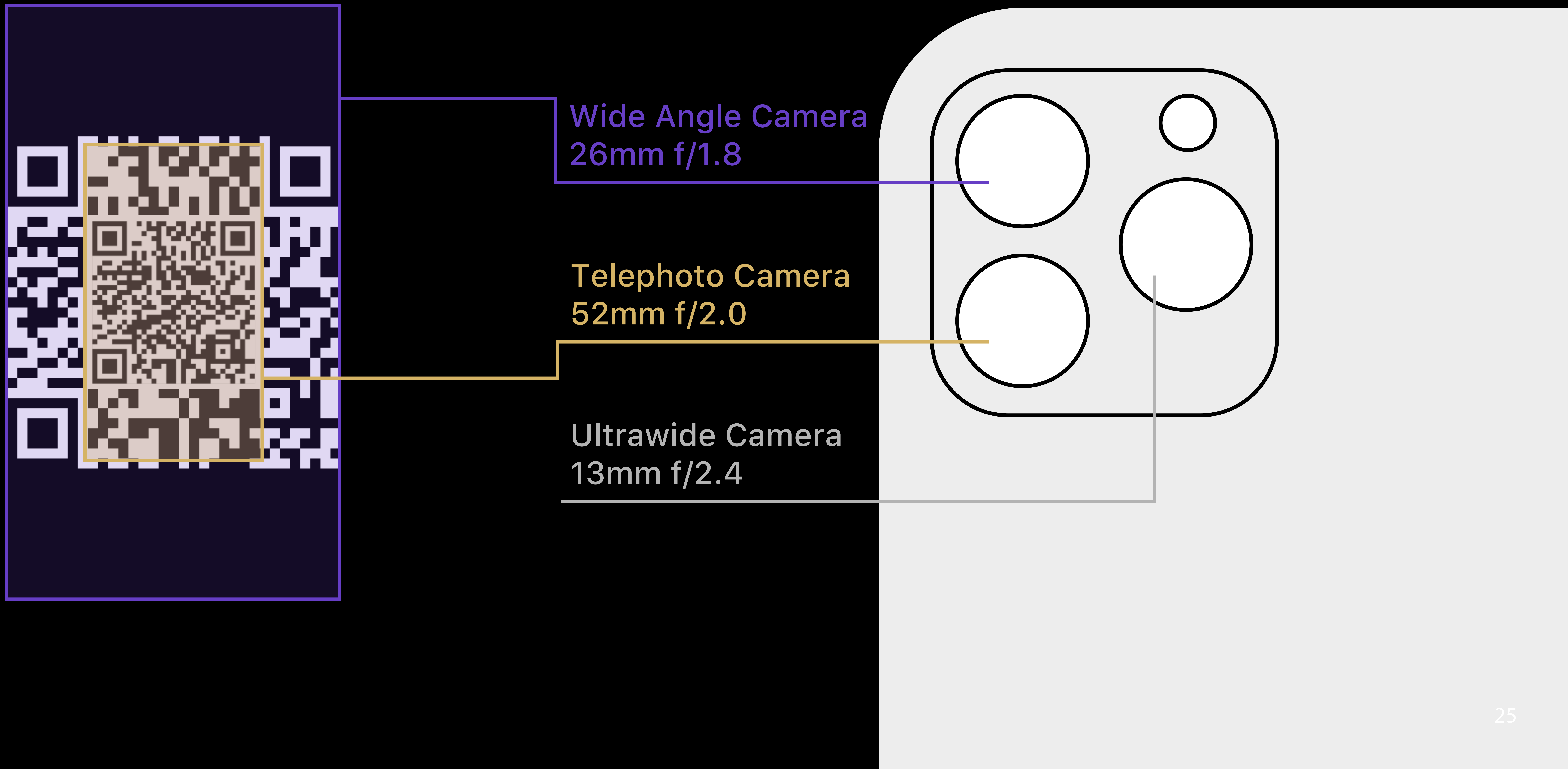
- Recent smartphones are equipped with multiple cameras
- Can we utilize them to further improve the throughput?



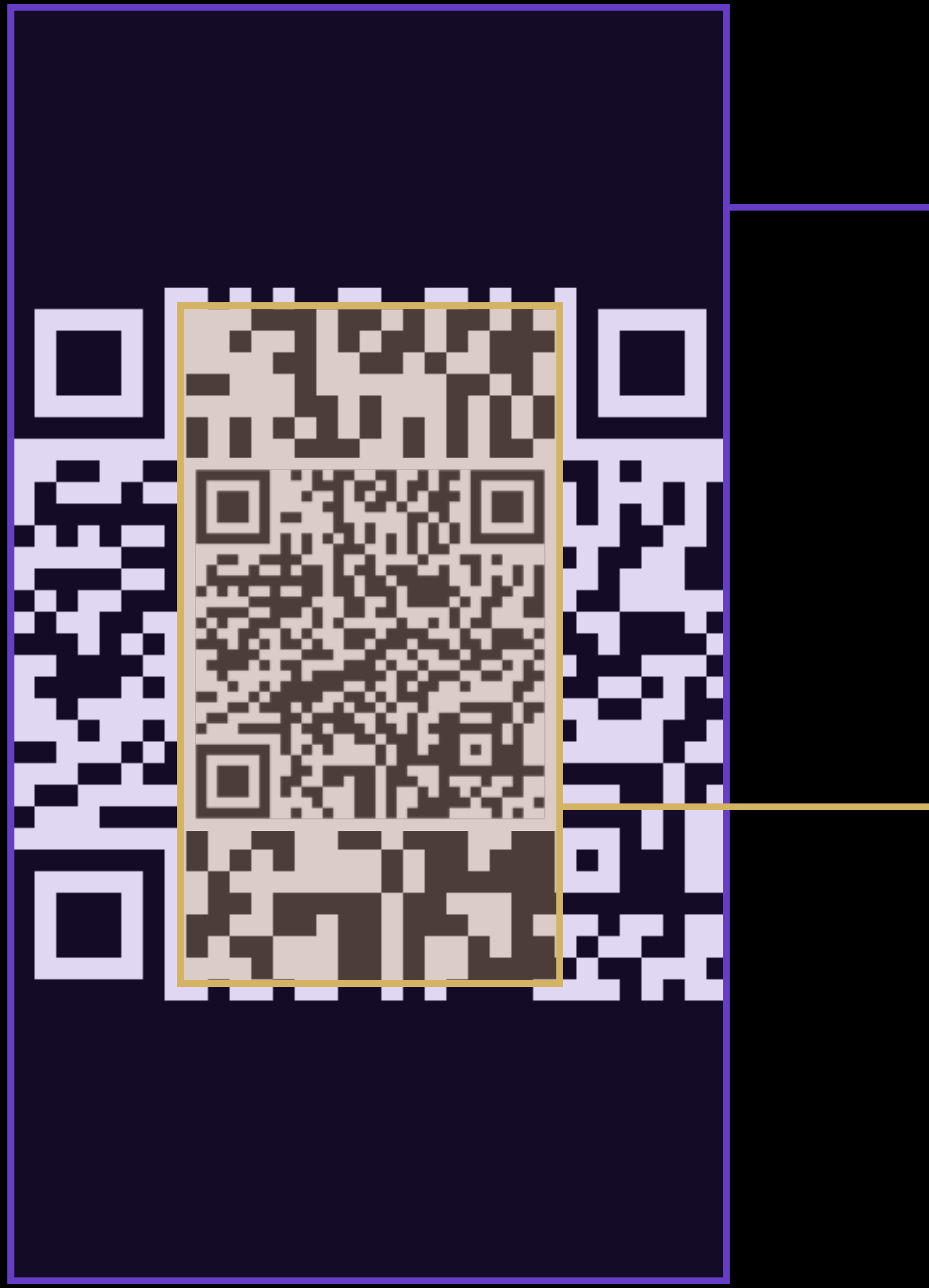
Nested Code Display Mode



Nested Code Display Mode

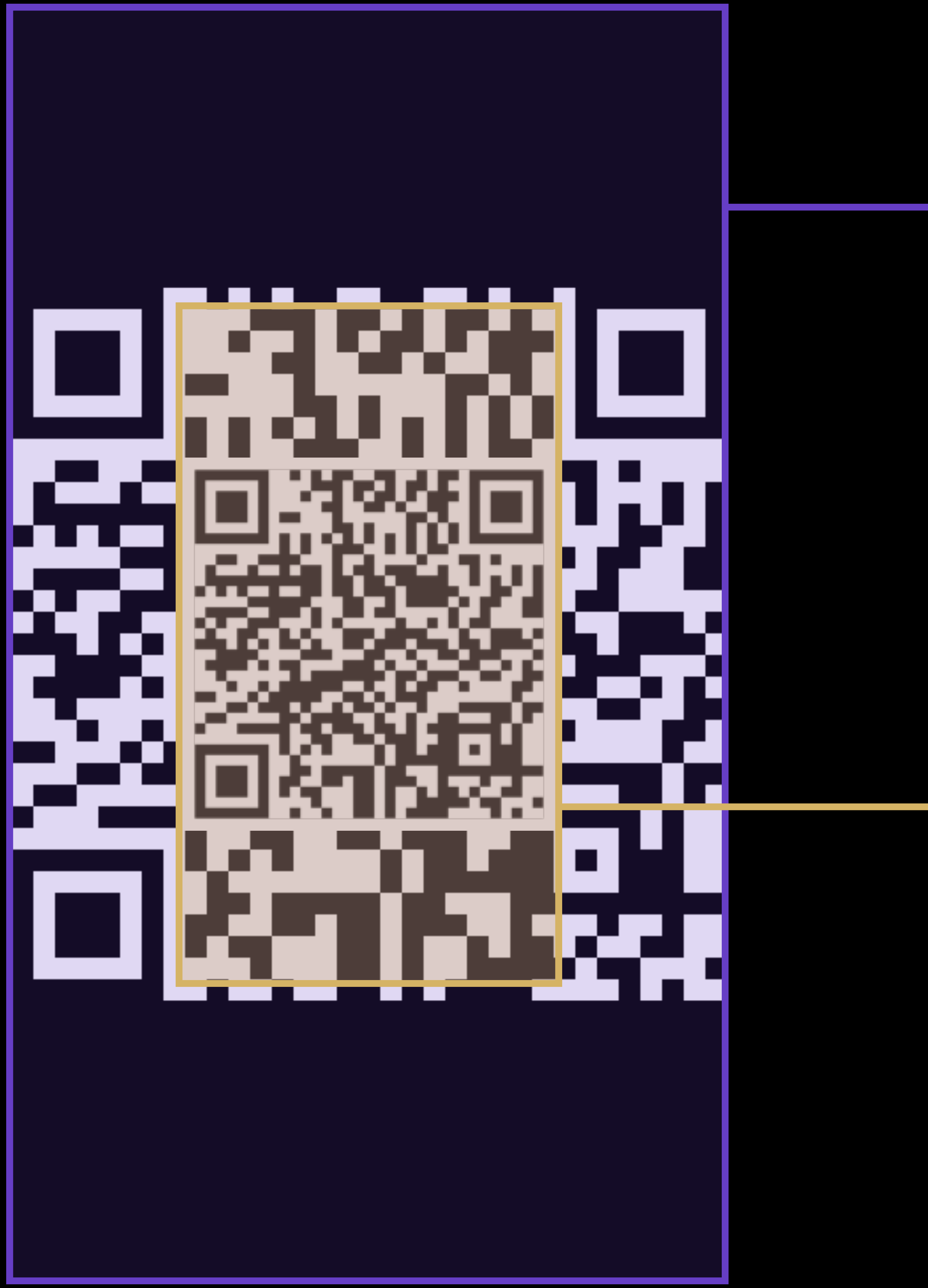


Nested Code Display Mode



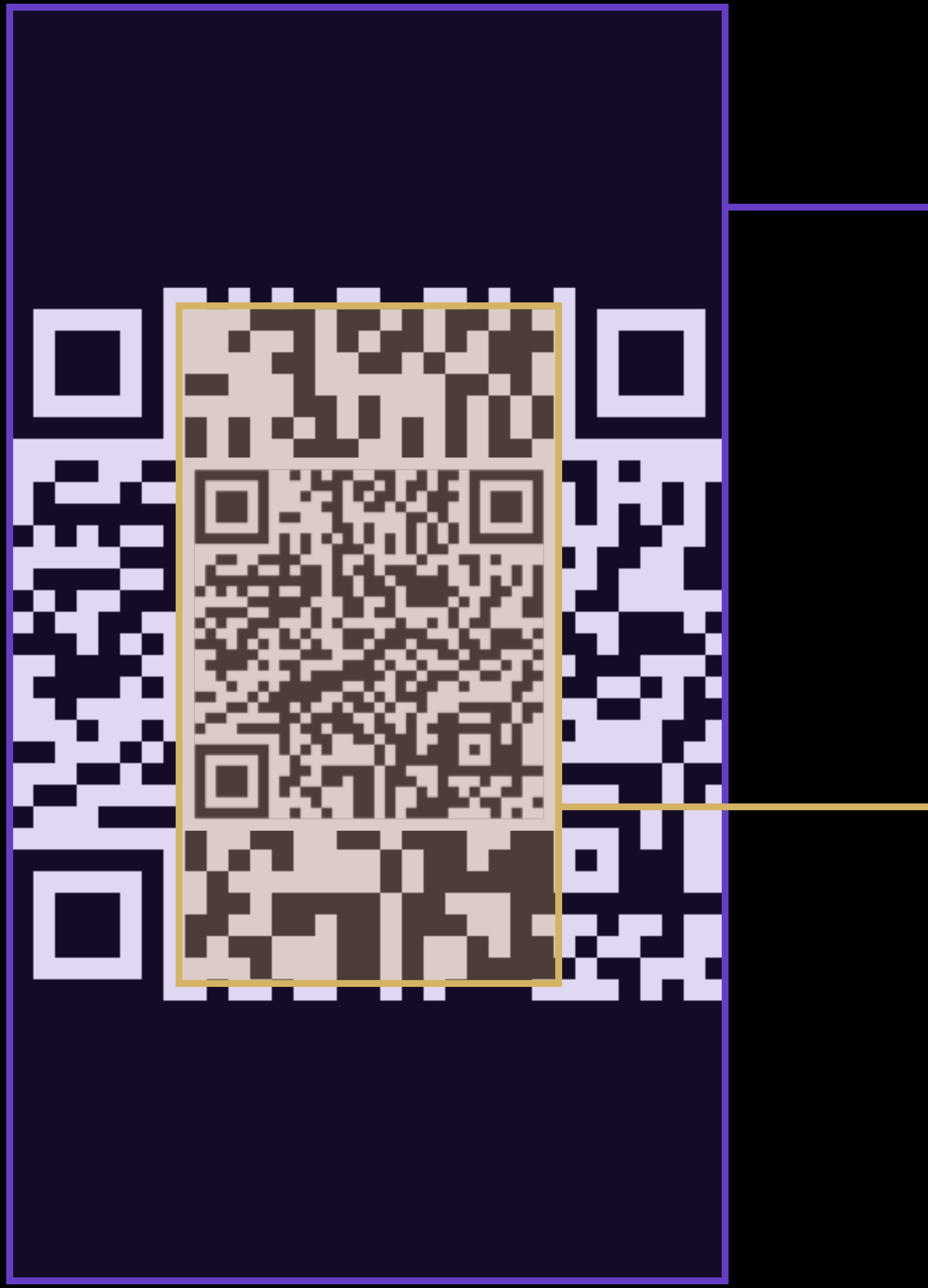
Nested Code Display Mode

- Ultrawide camera cannot be used
    - Resolving power
    - Distortion



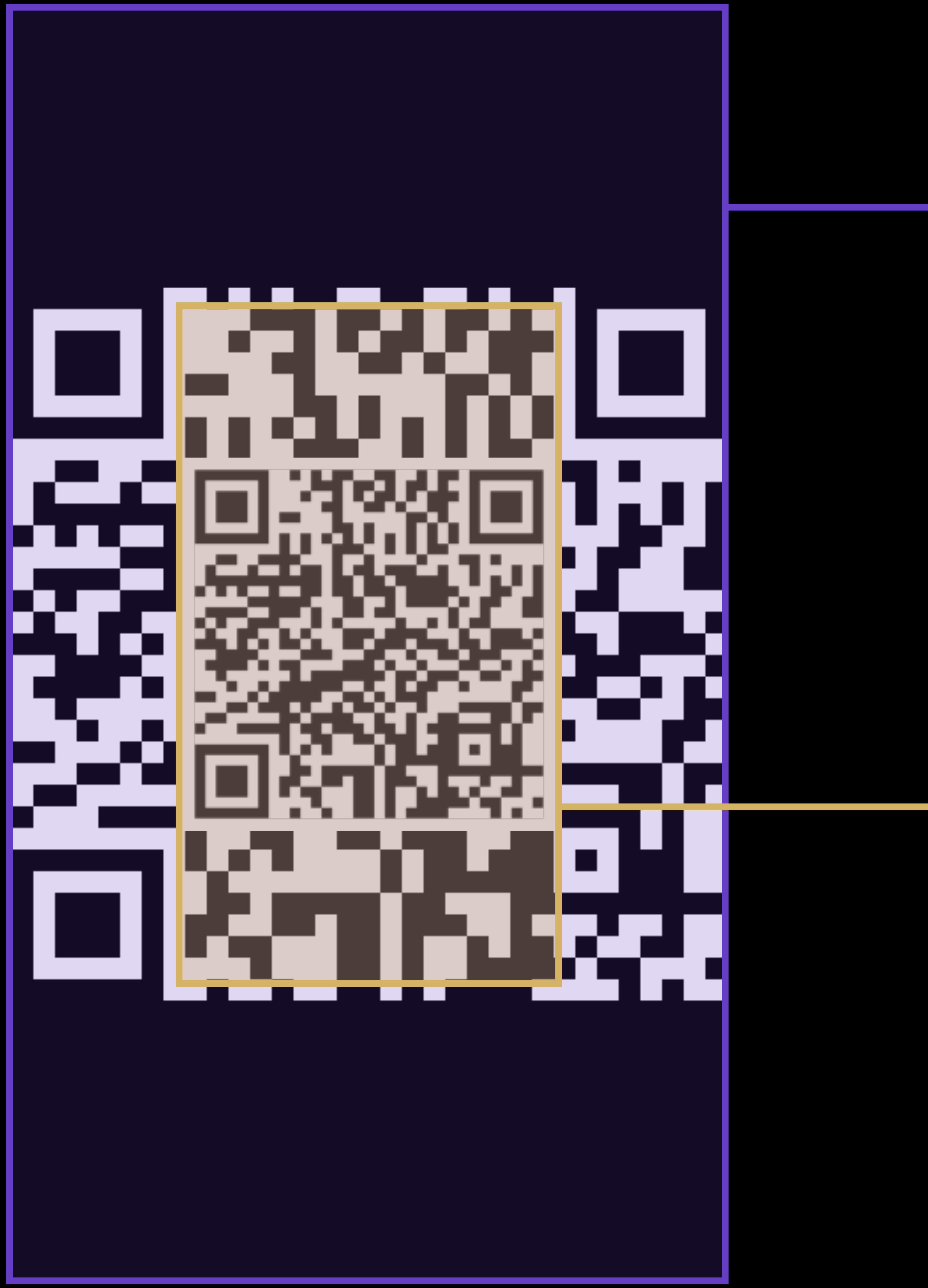
Nested Code Display Mode

- Ultrawide camera cannot be used
  - Resolving power
  - Distortion
- Ideally, 75% improvement in capacity
  - Only ~20% in practice



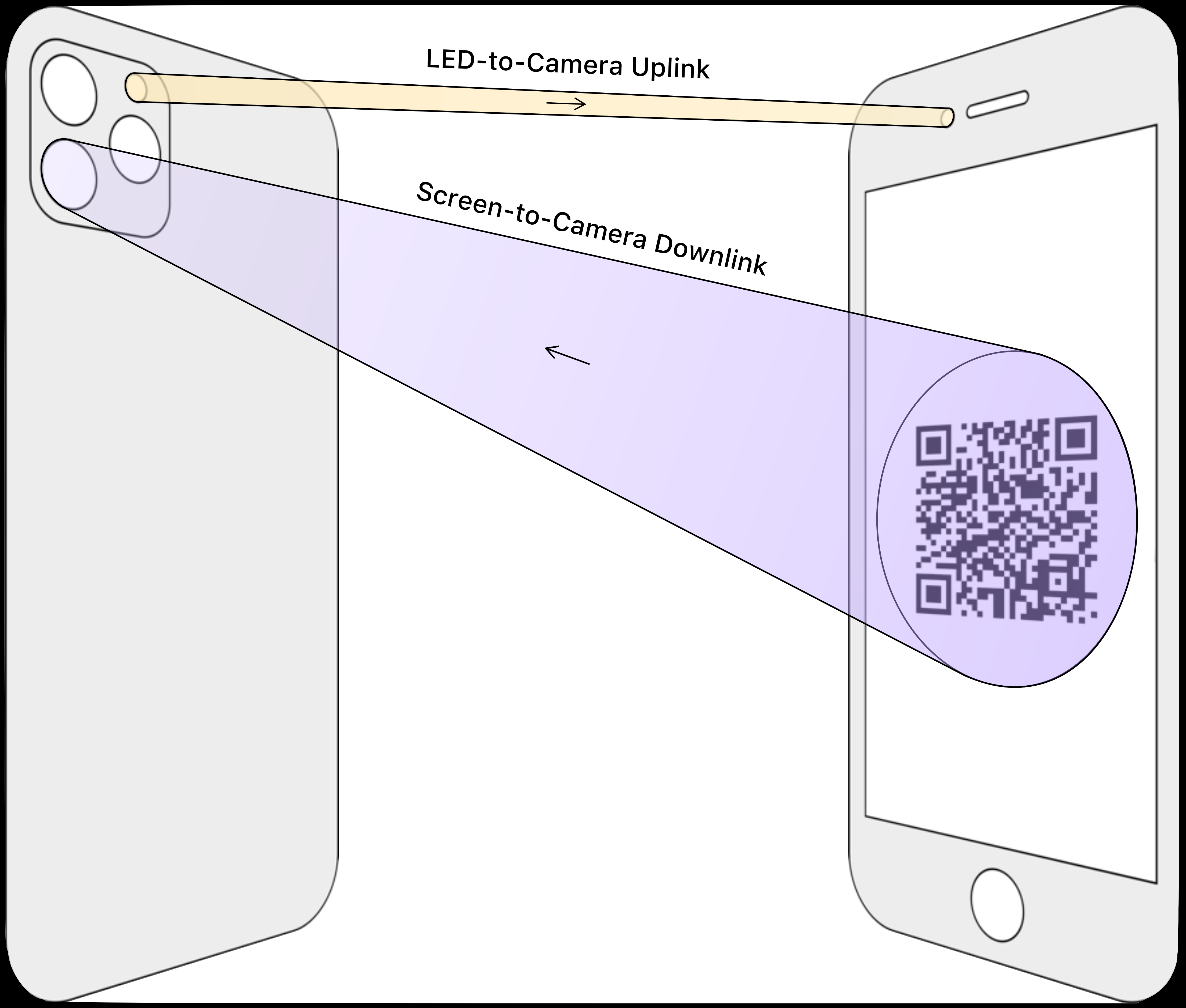
Nested Code Display Mode

- Ultrawide camera cannot be used
  - Resolving power
  - Distortion
- Ideally, 75% improvement in capacity
  - Only ~20% in practice
- Using dual camera is computationally expensive
  - Heating
  - Battery drain

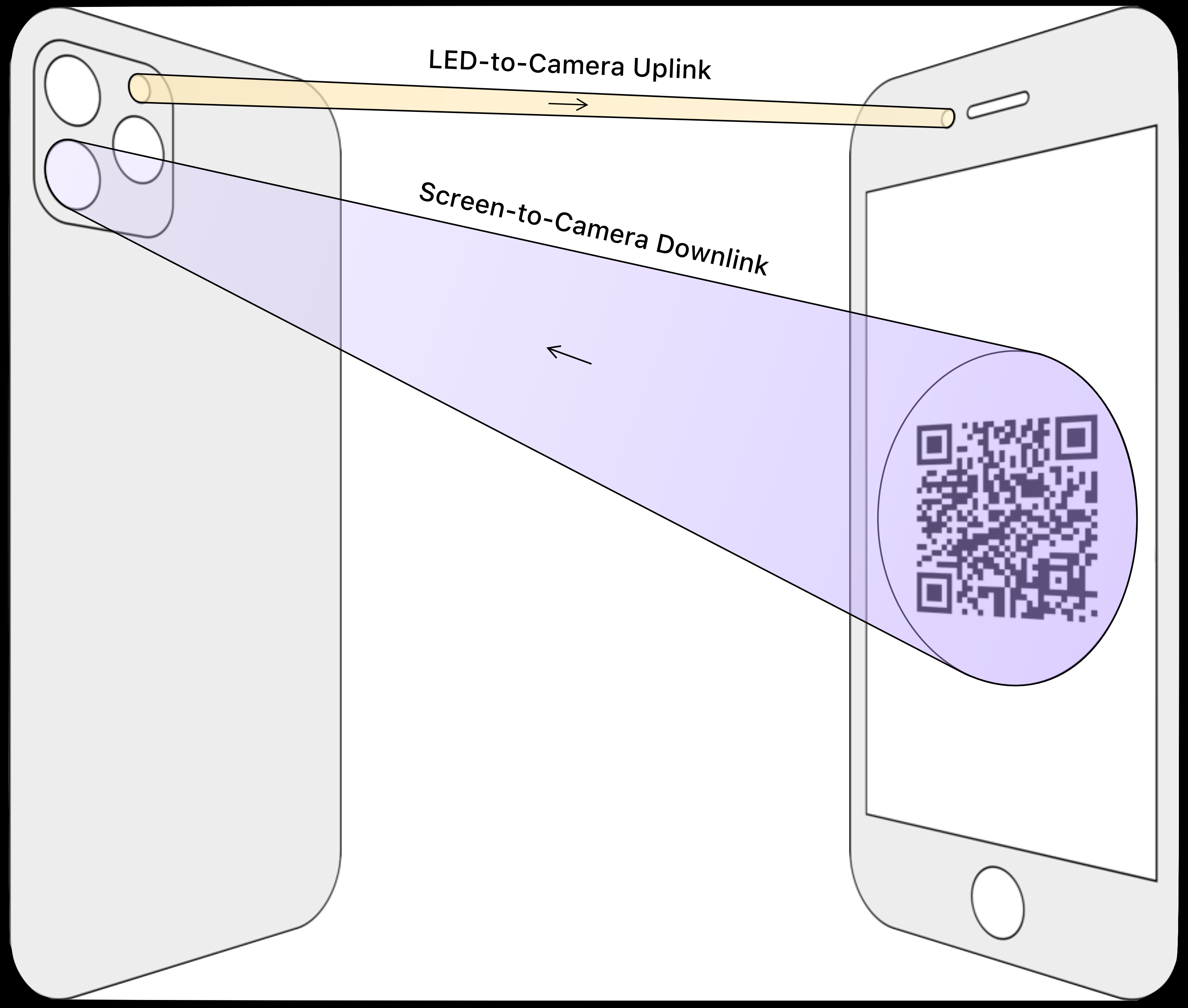


Duplex System

Duplex System

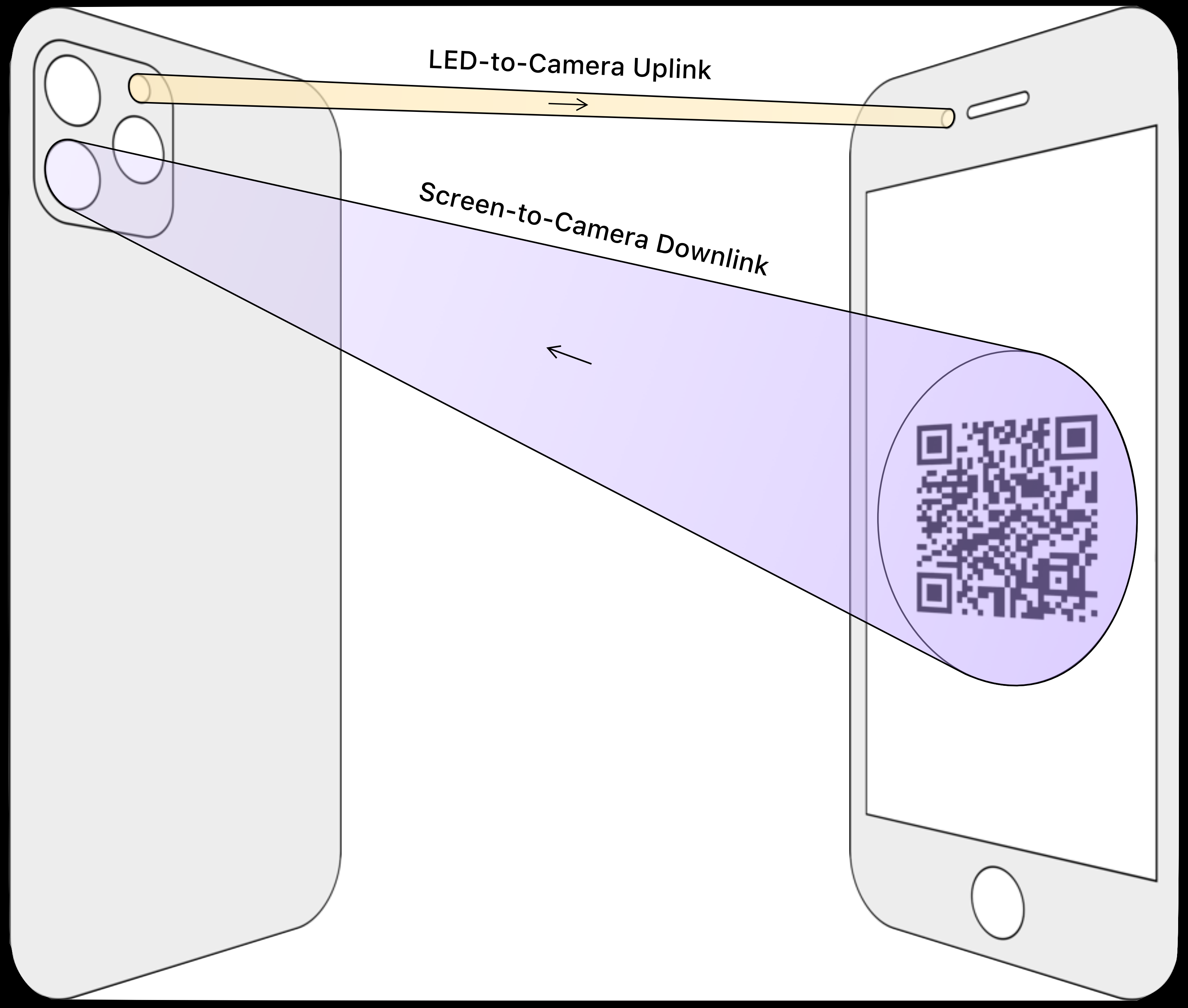


Duplex System



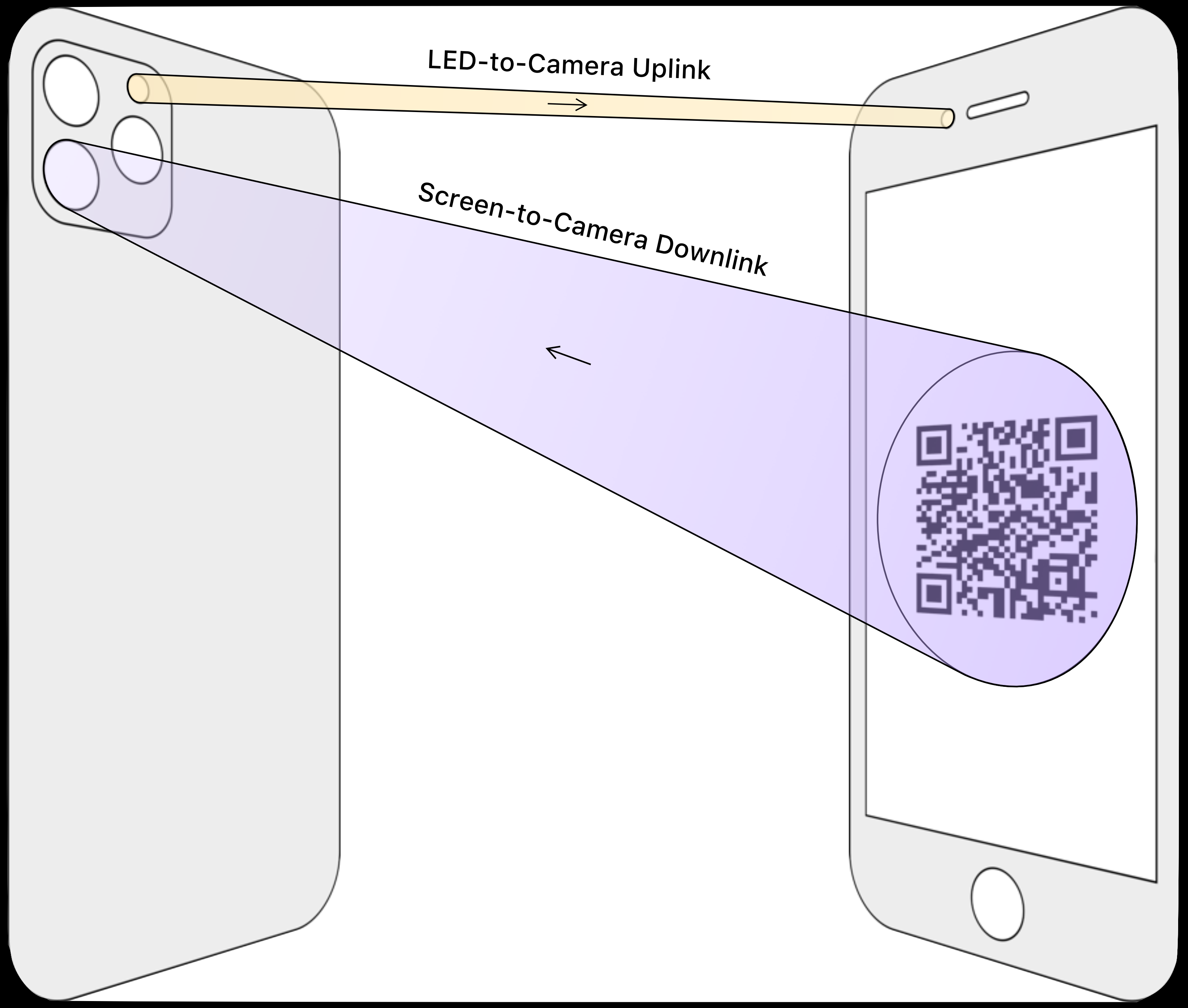
Duplex System

- Uplink uses on-off keying (OOK)
  - Number of white pixels in video frame as a signal
  - Detect rising edge



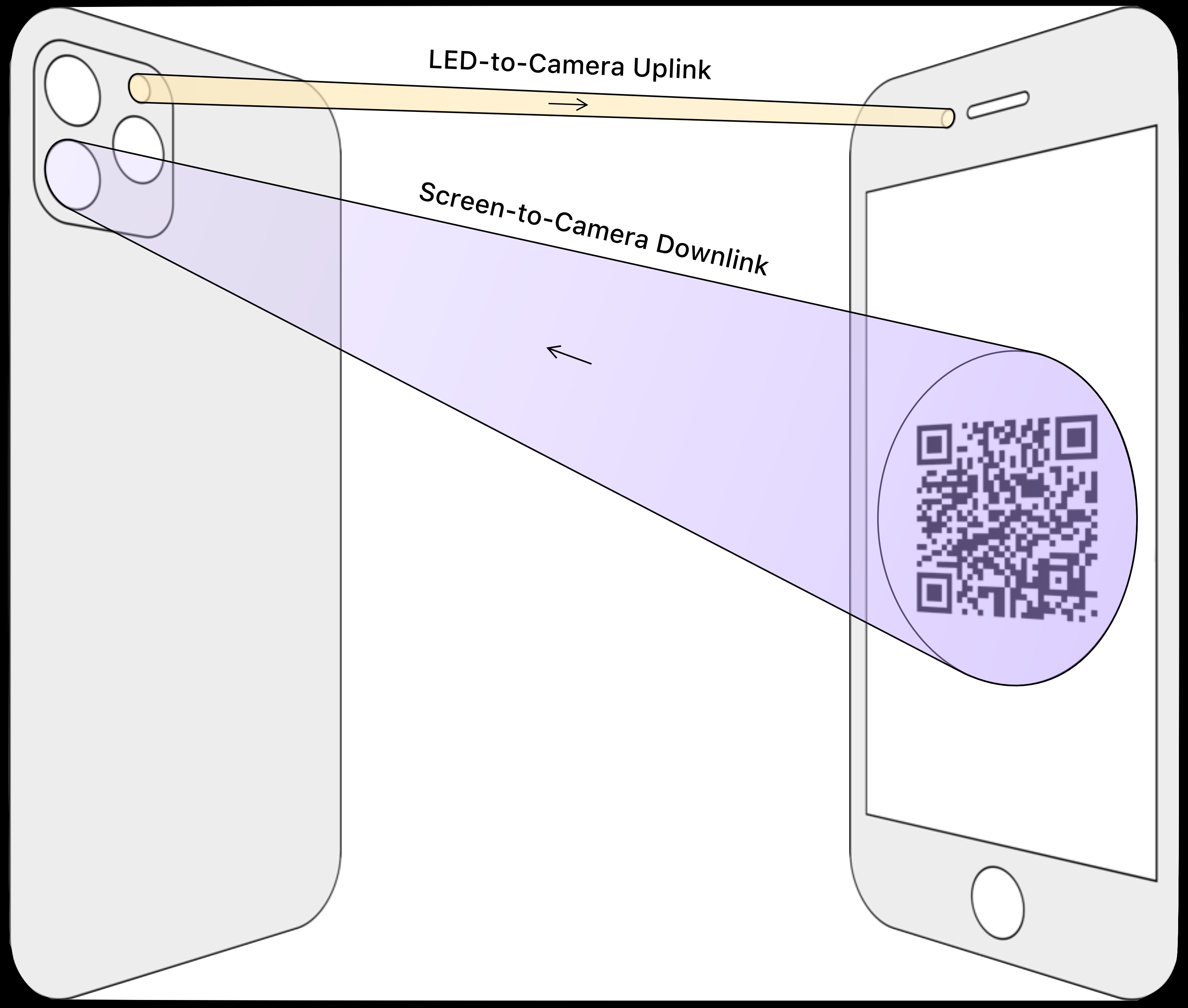
Duplex System

- Uplink uses on-off keying (OOK)
  - Number of white pixels in video frame as a signal
  - Detect rising edge
- Uplink has very limited throughput
  - 1 bit per frame
  - Difficult for implementing retransmission

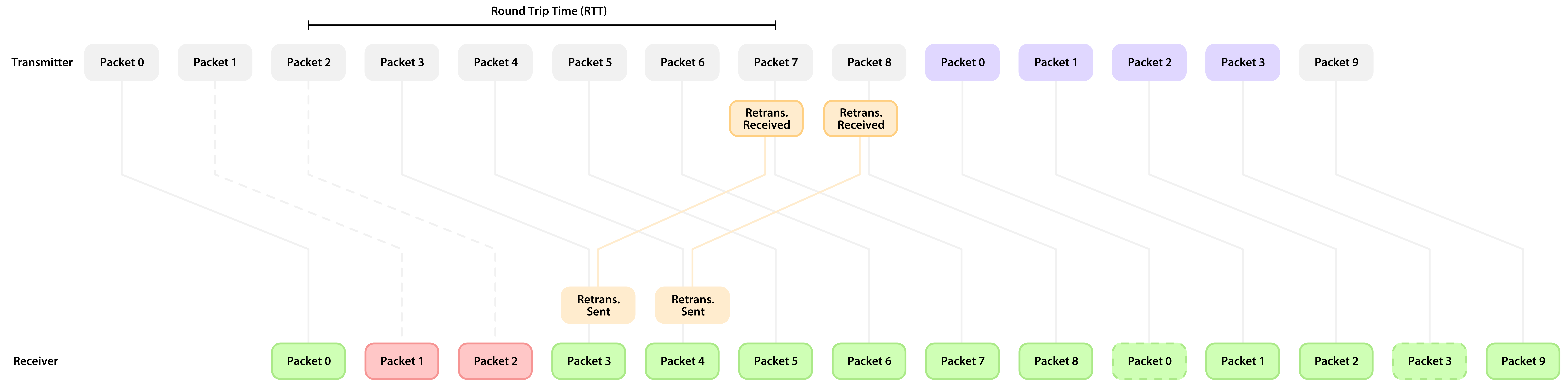


Duplex System

- Uplink uses on-off keying (OOK)
  - Number of white pixels in video frame as a signal
  - Detect rising edge
- Uplink has very limited throughput
  - 1 bit per frame
  - Difficult for implementing retransmission
- Long delay
  - Round-trip time: 200-400 ms



Retransmission Protocol



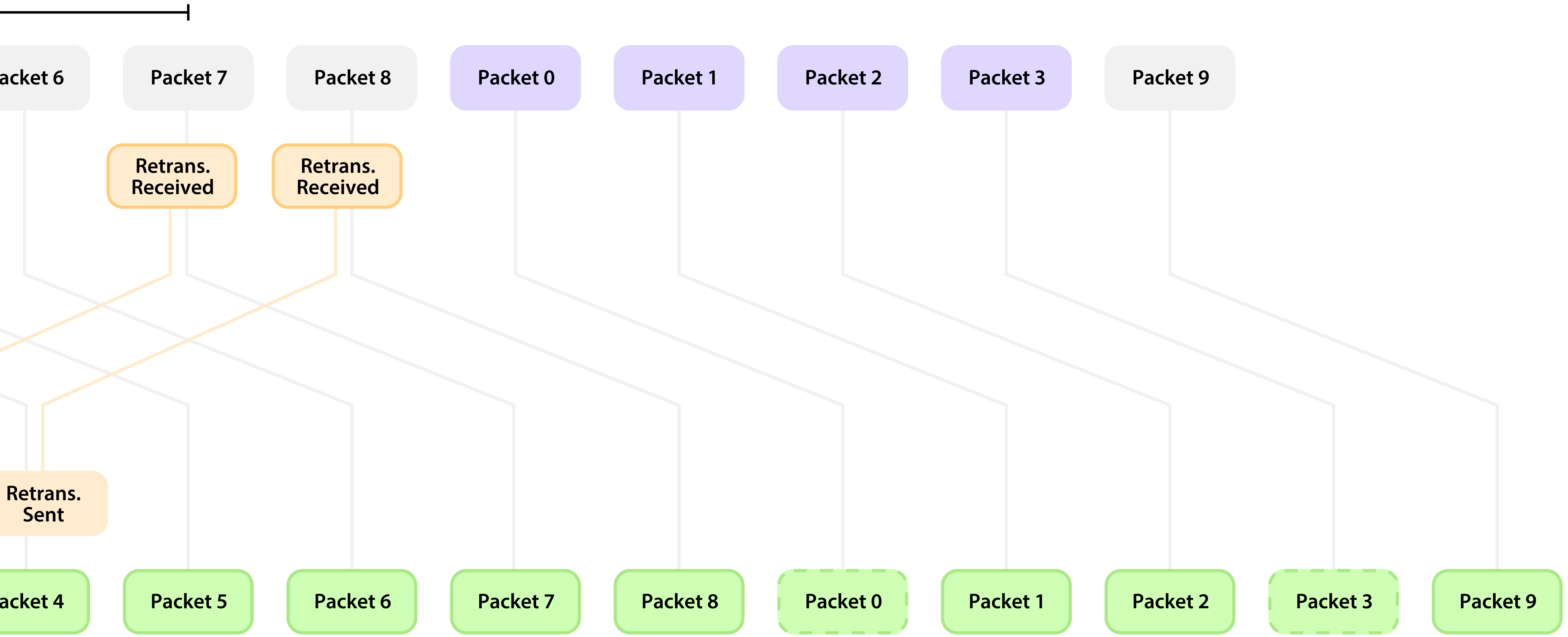
Retransmission Protocol

Round Trip Time (RTT)
Packet 0
Packet 1
Packet 2
Packet 3
Packet 4
Packet 5
Packet 6
Packet 7
Packet 8
P
Retrans. Received
Retrans. Received
Retrans. Sent
Retrans. Sent
Packet 0
Packet 1
Packet 2
Packet 3
Packet 4
Packet 5
Packet 6
P

Retransmission Protocol

Round Trip Time (RTT)
Packet 3
Packet 4
Packet 5
Packet 6
Packet 7
Packet 8
Packet 0
Packet 1
Packet 2
Packet 3
Retrans. Received
Retrans. Received
Retrans. Sent
Retrans. Sent
Packet 1
Packet 2
Packet 3
Packet 4
Packet 5
Packet 6
Packet 7
Packet 8
Packet 0
Packet 1

Retransmission Protocol



Experimental Results

All experiments were using an iPhone 11 Pro as the receiver and an iPad Pro (12.9 inch, 2018) as the transmitter.

Both devices were running iOS 13.6 and using the same version of *Prometheus* across the experiments.

The two devices were put on a table with a 25 cm separation from the iPhone's camera to the iPad's screen.

The exposure and focus of the devices' cameras are locked once the transmission starts.

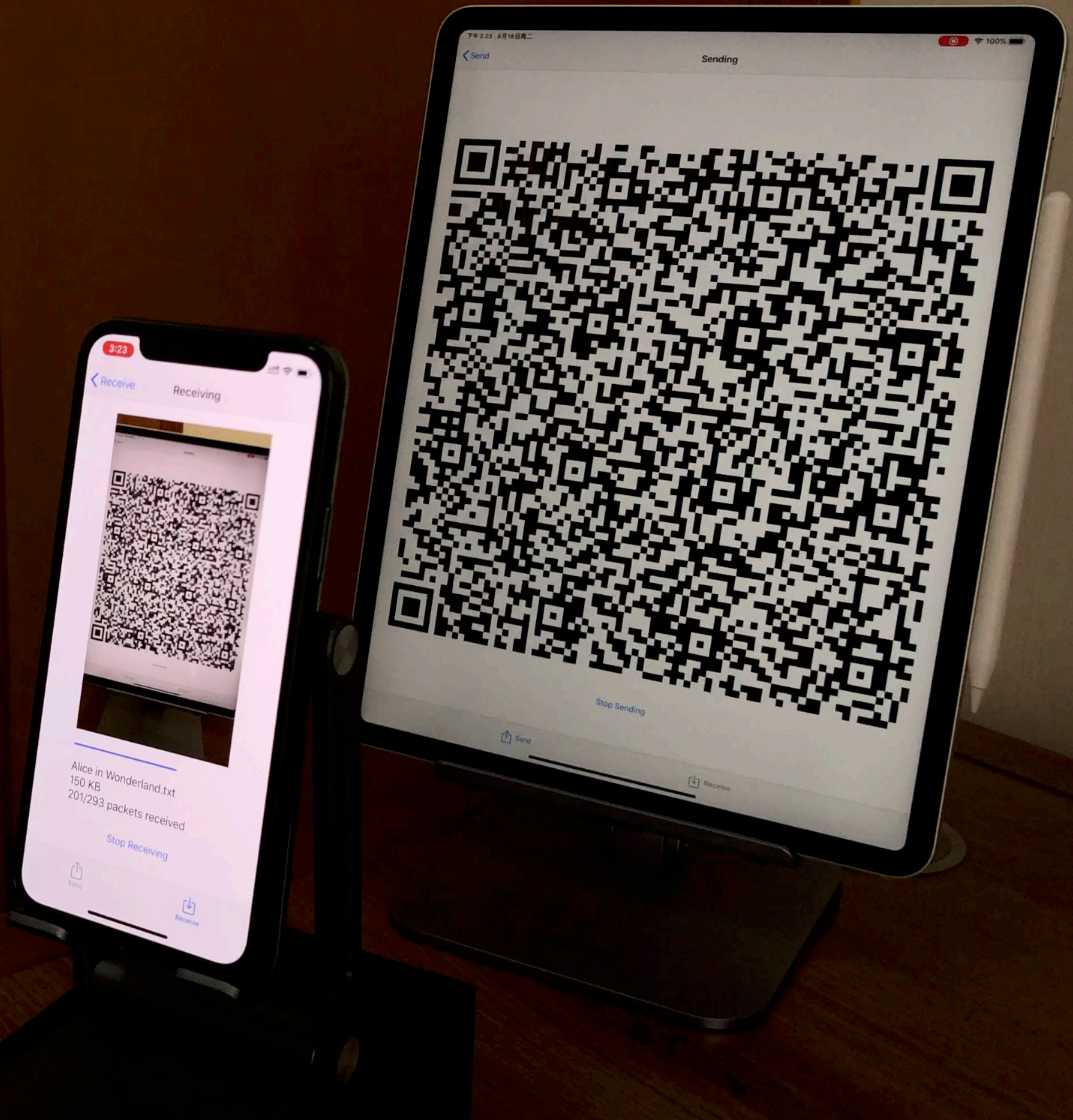
The payload of all experiments were a 150 kB text file "Alice in Wonderland.txt".

All experiments were repeated five times.

Unless explicitly specified, the following parameters were used in all the experiments:

- Simplex mode
- Code display mode: alternating code display
- Code display frame rate: 60 fps
- Code version: 13
- Code error correction level: low
- Receiver video resolution: 1280 by 720 pixels
- Receiver video frame rate: 60 fps

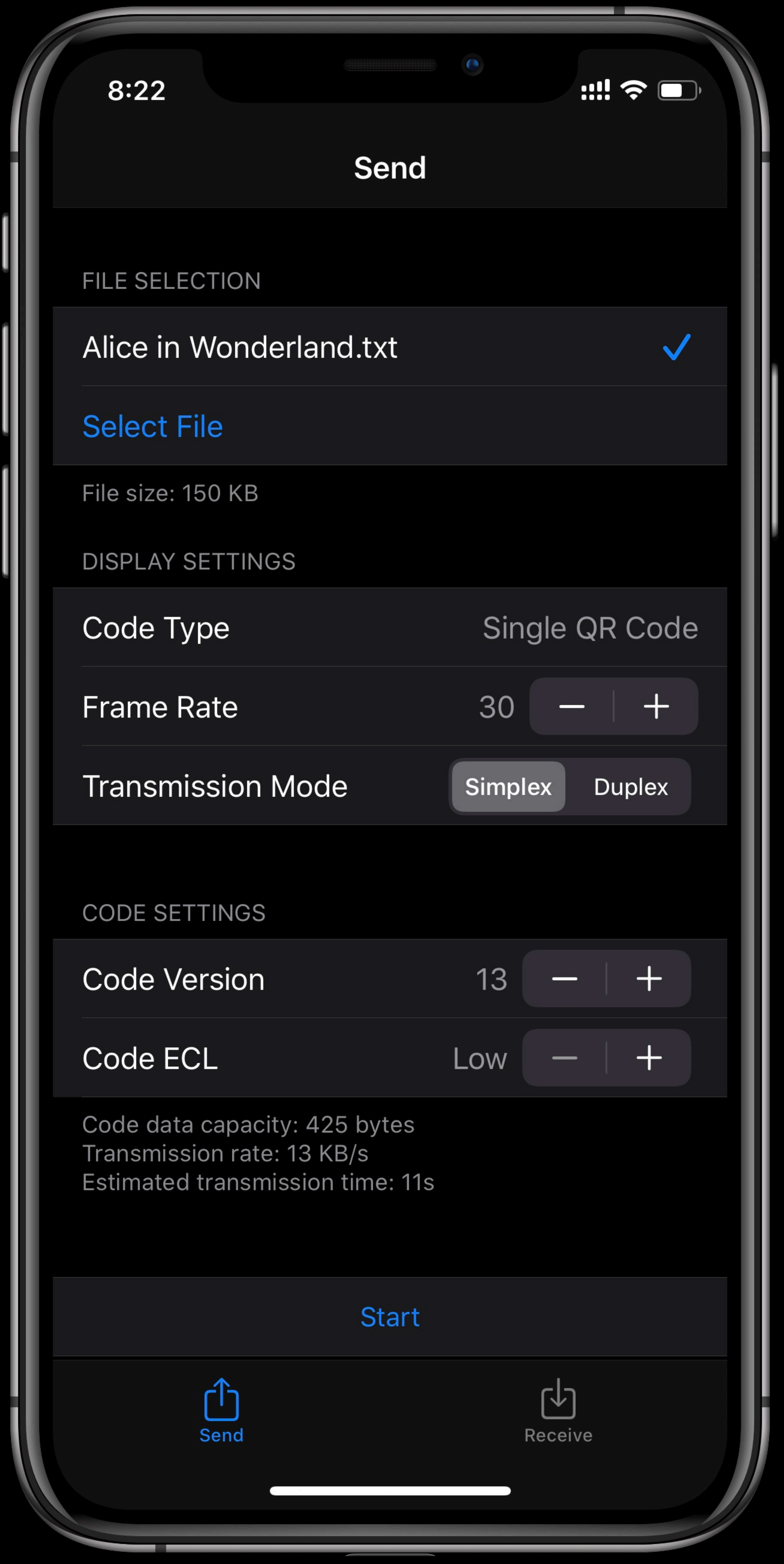
Setup



Receive
Receiving
Alice in Wonderland.txt
150 KB
201/293 packets received
Stop Receiving
Send
Sending
Stop Sending

UI: Transmitter Settings

- File to send
- Code display type
- Code display frame rate
- Simplex/duplex
- Code version
- Code error correction level (ECL)



8:22

Send

FILE SELECTION

Alice in Wonderland.txt ✓

Select File

File size: 150 KB

DISPLAY SETTINGS

Code Type	Single QR Code
Frame Rate	30 − +
Transmission Mode	[✓ Simplex] Duplex

CODE SETTINGS

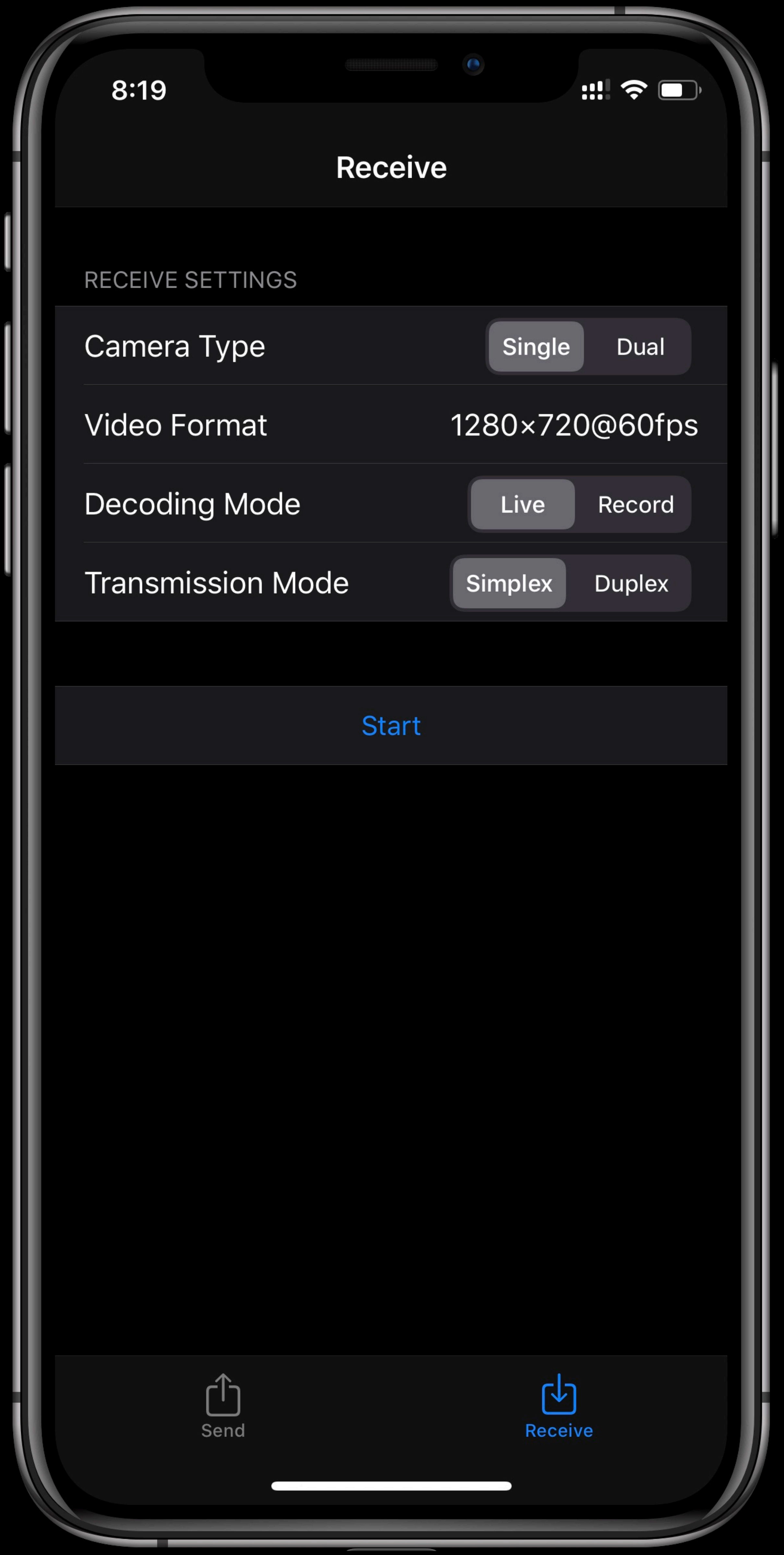
Code Version	13 − +
Code ECL	Low − +

Code data capacity: 425 bytes
Transmission rate: 13 KB/s
Estimated transmission time: 11s

Start

UI: Receiver Settings

- Camera Type
- Video Format
- Decoding Mode
- Simplex/duplex


8:19

Receive

RECEIVE SETTINGS

Camera Type	[✓ Single] Dual
Video Format	1280×720@60fps
Decoding Mode	[✓ Live] Record
Transmission Mode	[✓ Simplex] Duplex

Start

Experimental Results

Code Version	Lost Packet Percentage	Throughput (kbps)
13	0.0%	204.0
14	3.9%	211.3
15	22.4%	193.8
16	25.5%	209.6

Experimental Results

Code Version	Error Correction Level	Lost Packet Percentage	Throughput (kbps)
16	Low	25.5%	209.6
18	Medium	16.4%	224.8
22	Quartile	20.6%	215.3
25	High	21.4%	223.7

Code data capacity was kept nearly the same in this experiment.

Experimental Results

Code Version	Video Resolution	Lost Packet Percentage	Throughput (kbps)
28	1920×1080	15.6%	289.0
28	1280×720	42.6%	196.4
28	640×480	47.5%	179.8

Experimental Results

Code Display Mode	Lost Packet Percentage	Throughput (kbps)
Alternating	0.0%	204.0
Normal	42.5%	117.3

Experimental Results

Code Display Mode	Code Versions	Code ECLs	Packet Loss Rate	Throughput (kbps)
Nested	13; 10	Quartile; Low	0.0%	122.9
Normal	13	Low	0.0%	102.0

Code display frame rate was set to 30 fps in this experiment.

The displayed side length of the smaller code was 0.3 times that of the larger code.

Experimental Results

Dropped Packet Numbers	Retransmitted Packet Numbers
78	77, 78, 79
154	153, 154, 155
225	224, 225, 226
311, 312	310, 311, 312, 313

Both frame rates were set to 30 fps in this experiment.

Duplex mode was enabled in this experiment.

To generate packet loss, an obstacle was used to obscure part of the displayed code a few times.

Conclusion

Conclusion

- Literature reviews

Conclusion

- Literature reviews
- Proposed and implemented:
  - Prometheus
  - Alternating code display mode
  - Nested code display mode
  - Duplex mode and retransmission protocol

Conclusion

- Literature reviews
- Proposed and implemented:
    - Prometheus
    - Alternating code display mode
    - Nested code display mode
    - Duplex mode and retransmission protocol
- Experiments

Conclusion

- Literature reviews
- Proposed and implemented:
    - Prometheus
    - Alternating code display mode
    - Nested code display mode
    - Duplex mode and retransmission protocol
- Experiments
- Demo video & GitHub readme

Thank You

Q&A