🤖 JETSON NANO EMOTION DISPLAY SYSTEM

Fast, responsive emotion display system for Android tablets connected to robots.

📋 OVERVIEW

This system subscribes to emotion detection topics and displays emotions on Android tablets with:

- ⚡ **Ultra-fast response** (100ms update rate)
- ⏱️ **1-minute emotion display** (then returns to normal)
- 📱 **Tablet-optimized** web interface
- 🔄 **Auto-refresh** and **no delay**
- 🎯 **Production-ready** for Jetson Nano

🚀 QUICK INSTALLATION

Step 1: Download and Run Installer

bash

```
# Download the installer script
wget https://raw.githubusercontent.com/your-repo/install_emotion_system.sh

# Make executable
chmod +x install_emotion_system.sh

# Run installer (takes 5-10 minutes)
./install_emotion_system.sh
```

Step 2: Start System

bash

```
# Start everything
~/emotion_display_ws/start_emotion_system.sh
```

Step 3: Connect Tablet

1. **Connect tablet to same WiFi** as Jetson Nano
2. **Open browser** on tablet
3. **Go to:** `http://JETSON_IP:8000`
4. **Done!** 🎉

📁 FILE STRUCTURE

```
emotion_display_ws/
├── src/emotion_display/
│   ├── scripts/
│   │   └── emotion_display_node.py     # Main emotion node
│   ├── launch/
│   │   └── emotion_display.launch      # Launch file
│   ├── test/
│   │   ├── test_emotion_publisher.py   # Test emotion sender
│   │   ├── test_network.py            # Network tester
│   │   └── run_complete_test.sh       # Complete test
│   ├── package.xml                    # ROS package config
│   └── CMakeLists.txt                # Build config
├── start_emotion_system.sh            # System startup
└── QUICK_REFERENCE.txt               # Quick commands
```

🎭 HOW TO USE

Start the System

bash

```bash
# Method 1: Use startup script
~/emotion_display_ws/start_emotion_system.sh

# Method 2: Manual start
roscore &
sleep 2
roslaunch emotion_display emotion_display.launch
```

Send Emotions to Display

bash

```bash
# Send emotions via ROS topic
rostopic pub /emotion_detection std_msgs/String "data: 'happy'"
rostopic pub /emotion_detection std_msgs/String "data: 'sad'"
rostopic pub /emotion_detection std_msgs/String "data: 'angry'"
rostopic pub /emotion_detection std_msgs/String "data: 'surprised'"
rostopic pub /emotion_detection std_msgs/String "data: 'joy'"
rostopic pub /emotion_detection std_msgs/String "data: 'fear'"
rostopic pub /emotion_detection std_msgs/String "data: 'normal'"
```

Connect Your Emotion Detection

python

```python
#!/usr/bin/env python3
import rospy
from std_msgs.msg import String

# Your emotion detection code here...
# When you detect an emotion:

pub = rospy.Publisher('/emotion_detection', String, queue_size=1)
msg = String()
msg.data = "happy"  # or whatever emotion you detected
pub.publish(msg)
```

📱 TABLET INTERFACE

URLs

- **Main Display:** `http://JETSON_IP:8000`
- **Status Page:** `http://JETSON_IP:8000/status`
- **API Endpoint:** `http://JETSON_IP:8000/emotion`

Features

- 🎨 **Full-screen emotion display**
- ⏱️ **Countdown timer** (shows time remaining)
- 🔄 **Auto-refresh** every 100ms
- 📊 **Status indicators**
- 💡 **Screen wake-lock** (prevents sleep)
- ⚡ **Flash effect** for new emotions

🧪 TESTING

Complete System Test

bash

```bash
cd ~/emotion_display_ws/src/emotion_display/test
./run_complete_test.sh
```

Test Network Only

bash

```bash
python3 test_network.py
```

Test Emotions Only

```bash
# Test specific emotion
python3 test_emotion_publisher.py happy

# Test all emotions (cycles through each)
python3 test_emotion_publisher.py
```

Manual Testing

```bash
# Check if system is running
rostopic list | grep emotion_detection
curl http://localhost:8000/status

# Send test emotion
rostopic pub /emotion_detection std_msgs/String "data: 'joy'"

# Check response
curl http://localhost:8000/emotion
```

⚙️ CONFIGURATION

Change Settings

Edit the launch file: `~/emotion_display_ws/src/emotion_display/launch/emotion_display.launch`

```xml
<launch>
    <!-- Change these parameters -->
    <arg name="emotion_topic" default="/emotion_detection" />
    <arg name="http_port" default="8000" />

    <node pkg="emotion_display" type="emotion_display_node.py" name="emotion_display_n
        <param name="emotion_topic" value="$(arg emotion_topic)" />
        <param name="http_port" value="$(arg http_port)" />
    </node>
</launch>
```

Parameters

- `emotion_topic`: ROS topic to subscribe to (default: `/emotion_detection`)
- `http_port`: HTTP server port (default: `8000`)
- `emotion_duration`: How long emotions display (default: 60 seconds)

🎨 SUPPORTED EMOTIONS

Emotion	Color	Emoji	Duration
`happy`	Green	😊	60 seconds
`sad`	Blue	😢	60 seconds
`angry`	Red	😠	60 seconds
`surprised`	Yellow	😲	60 seconds
`joy`	Orange	😄	60 seconds
`fear`	Purple	😨	60 seconds
`disgust`	Lime	🤢	60 seconds
`normal`	Gray	🤖	Always

Note: After 60 seconds, any emotion automatically returns to `normal`.

🔧 TROUBLESHOOTING

System Won't Start

```bash
# Check ROS installation
roscore
# Should start without errors

# Check workspace build
cd ~/emotion_display_ws
catkin_make

# Check dependencies
sudo apt-get install ros-noetic-cv-bridge python3-opencv
```

Tablet Can't Connect

bash

```bash
# Check IP address
hostname -I

# Test HTTP server
curl http://localhost:8000/status

# Check firewall
sudo ufw allow 8000
```

No Emotions Showing

bash

```bash
# Check if topic exists
rostopic list | grep emotion

# Check if messages are being sent
rostopic echo /emotion_detection

# Send test emotion
rostopic pub /emotion_detection std_msgs/String "data: 'test'"
```

Slow Response

bash

```bash
# Check system resources
htop

# Restart system
~/emotion_display_ws/start_emotion_system.sh
```

📊 MONITORING

Check System Status

bash

```
# ROS nodes
rosnode list

# Topics
rostopic list
rostopic hz /emotion_detection

# HTTP server
curl http://localhost:8000/status | python3 -m json.tool
```

Logs

bash

```
# ROS logs
rosnode info emotion_display_node

# System logs
journalctl -f | grep emotion
```

🔄 INTEGRATION EXAMPLES

With OpenCV Emotion Detection

python

```python
#!/usr/bin/env python3
import rospy
from std_msgs.msg import String
import cv2
# Your emotion detection imports...

def detect_and_publish():
    rospy.init_node('emotion_detector')
    pub = rospy.Publisher('/emotion_detection', String, queue_size=1)

    # Your emotion detection code
    cap = cv2.VideoCapture(0)

    while not rospy.is_shutdown():
        ret, frame = cap.read()

        # Your emotion detection here
        emotion = detect_emotion(frame)  # Your function

        if emotion:
            msg = String()
            msg.data = emotion
            pub.publish(msg)
            rospy.loginfo(f"Detected emotion: {emotion}")

        rospy.sleep(0.1)  # 10Hz
```

With Voice Control

python

```python
#!/usr/bin/env python3
import rospy
from std_msgs.msg import String

def voice_emotion_control():
    pub = rospy.Publisher('/emotion_detection', String, queue_size=1)

    while True:
        emotion = input("Enter emotion (happy/sad/angry/normal): ")
        if emotion in ['happy', 'sad', 'angry', 'surprised', 'joy', 'fear', 'normal']:
            msg = String()
            msg.data = emotion
            pub.publish(msg)
            print(f"✅ Sent: {emotion}")
        else:
            print("❌ Invalid emotion")
```

🚀 AUTO-START ON BOOT

Create Service

bash

```bash
sudo nano /etc/systemd/system/emotion-display.service
```

ini

```ini
[Unit]
Description=Robot Emotion Display System
After=network.target

[Service]
Type=simple
User=jetson
WorkingDirectory=/home/jetson/emotion_display_ws
ExecStart=/home/jetson/emotion_display_ws/start_emotion_system.sh
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target
```

```bash
# Enable service
sudo systemctl enable emotion-display.service
sudo systemctl start emotion-display.service

# Check status
sudo systemctl status emotion-display.service
```

📞 SUPPORT

Common Issues

1. **"Unable to register with master node"** → Start `roscore` first
2. **"Connection refused"** → Check firewall and IP address
3. **"Slow response"** → Reduce update rate or check network
4. **"No module named cv_bridge"** → Install ROS dependencies

Get Help

```bash
# System info
~/emotion_display_ws/src/emotion_display/test/test_network.py

# Full test
~/emotion_display_ws/src/emotion_display/test/run_complete_test.sh
```

📄 LICENSE

MIT License - Feel free to use in your robot projects!

🎯 QUICK COMMANDS REFERENCE

bash

```bash
# Start system
~/emotion_display_ws/start_emotion_system.sh

# Test system
cd ~/emotion_display_ws/src/emotion_display/test && ./run_complete_test.sh

# Send emotion
rostopic pub /emotion_detection std_msgs/String "data: 'happy'"

# Check status
curl http://localhost:8000/status

# View tablet
# Go to: http://JETSON_IP:8000
```

🎉 **That's it! Your robot emotions should now display instantly on your Android tablet!**