Rotinas de Movimento - Plataforma de Stewart

Visão Geral

`routines.html` centraliza a execução de movimentos automatizados (senoides, círculos, hélices etc.), além de monitorar e gravar as trajetórias comandadas x reais. O script `motion-logic.js` integra presets configuráveis, Chart.js com zoom/pan, IndexedDB para logging e o backend FastAPI (`/motion/*`) que alimenta o ESP32.

Características

- Catálogo de presets com sliders dedicados para amplitude, frequência, offsets e duração.
- Start/Stop e painel de status com contagem regressiva, último comando enviado e progresso.
- Visualização em dois gráficos (setpoint e curso real) com zoom multitouch (Hammer.js + chartjs-plugin-zoom).
- Gravação local em IndexedDB + exportação em CSV com marca temporal e valores dos seis pistões.
- WebSocket dedicado que combina `motion_tick` (pose alvo) e telemetria normalizada para manter preview, status e chart sincronizados.
- Ferramentas de seleção rápida de pistões (liga/desliga por checkbox ou botões “Todos On/Off”).

Arquitetura

Backend (`interface/backend/app.py`)

- `POST /motion/start`: recebe `MotionRequest` com `routine`, `duration_s`, `hz`, parâmetros específicos (amp, ax, ay, z_amp_mm etc.) e inicia a `MotionRunner`.
- `POST /motion/stop`: interrompe a rotina atual e retorna o estado final.
- `GET /motion/status`: diz se há rotina ativa, progresso em % e presets atuais.
- `POST /serial/open|close|status`: gerenciamento da porta.
- WebSocket `/ws/telemetry`: envia mensagens `motion_tick` (com `pose_cmd`) + telemetria real (campos `Y`, `sp_mm`, `mpu`...).

Frontend (`routines.html` + `motion-logic.js`)

- `initCommonSerialControls()` padroniza botões de porta, LED e toasts.
- IndexedDB:
  - `initMotionDB()` cria `MotionTrajectoryDB.motion_data`.
  - `saveMotionDataBatch()` grava pacotes de telemetria em lotes (buffer de 10 registros / 500 ms).
- Chart.js:
  - `initMotionCharts()` instancia dois gráficos (Cmd e Real) com janelas deslizantes de 30 s.
  - `startMotionChart()/stopMotionChart()/clearMotionChart()` controlam a gravação.
  - `toggleMotionPistonVisibility()` atua em datasets específicos (cmd x real).
- Presets:
  - `MOTION_PRESET_CONFIG` descreve cada cartão (nome, rotina, parâmetros padrão).
  - `loadPresets()` injeta sliders e botões `Iniciar`.

    - `startMotionRoutine(card)` lê valores do card, monta o `MotionRequest` e chama `POST /motion/start`.
- WebSocket:
    - `setupMotionWebSocket()` escuta `window.ws` (já criado por `common.js`). Marca `window.ws._motionHandlerAttached` para evitar handlers duplicados.
    - `handleMotionTelemetry()` divide `pose_cmd` e telemetria real, atualiza charts e UI.

Painéis e Controles

1. **Status da Conexão / Serial** – herdado do layout padrão (select de porta, conectar/desconectar, botão "Atualizar").
2. **Cards de Rotina** – cada preset mostra ícone, descrição, sliders (amp, Hz, duração etc.) e botão "Iniciar".
3. **Botões Globais** – "Parar Rotina", indicadores de modo manual/automático, tempo restante e últimos comandos.
4. **Painel do Gráfico** – duas abas (Comando x Real) compartilhando botões: Iniciar gravação, Parar, Limpar, Exportar CSV, Reset Zoom x2.
5. **Seleção de Pistões** – toggles individuais e botões "Todos ON/OFF" separados para curvas de comando e reais.

Fluxo de Funcionamento

1. Ao abrir a página:
    - Serial controls são iniciados.
    - IndexedDB e gráficos são montados.
    - `loadPresets()` cria os cards.
    - `setupMotionWebSocket()` aguarda mensagens de telemetria.
    - `checkMotionStatus()` verifica se existe rotina em progresso (útil após refresh).
2. Pressionou "Iniciar" em um card:
    - `startMotionRoutine` extrai parâmetros, monta `MotionRequest` e faz `fetch('/motion/start')`.
    - O backend valida limites, inicia `MotionRunner` e devolve status → UI atualiza o painel de progresso.
3. Enquanto a rotina roda:
    - WebSocket envia `motion_tick` com `pose_cmd` → usado para atualizar o gráfico de comando e preview 3D (caso outra aba esteja aberta).
    - Telemetria real (`Y`) atualiza o gráfico real, medidas numéricas e gravação no IndexedDB se o usuário clicou em "Iniciar gravação".
    - Heartbeat verifica se ainda chegam mensagens; em falha, tenta reconectar (`scheduleReconnect`).
4. "Parar Rotina" ou conclusão automática:
    - Chama `POST /motion/stop` e atualiza status.
    - O gráfico pode continuar gravando telemetria manualmente se desejado.

Passo a Passo de Uso

1. Levantar backend (`python app.py`) e abrir `routines.html`.
2. Conectar o ESP32 via seção "Conexão Serial".
3. Escolher um preset, ajustar sliders (amplitude, frequência, duração, offsets).

4. (Opcional) Preparar gravação clicando em “Iniciar Gravação” no painel de gráficos.
5. Apertar “Iniciar” no card desejado → acompanhar indicadores de status e o gráfico live.
6. A qualquer momento:
    - Clique em “Parar Rotina” para abortar.
    - Use zoom/pan do Chart.js para analisar detalhes.
    - Exporte CSV para comparar offline.

Configurações Avançadas

- **Novos presets**: adicione entradas ao objeto `MOTION_PRESET_CONFIG` (defina `title`, `routine`, `defaultParams` e trate no backend).
- **Número máximo de pontos**: ajuste `MAX_VISIBLE_POINTS` e `CHART_WINDOW_SECONDS` para janelas maiores/menores.
- **Buffer de IndexedDB**: `DB_BATCH_SIZE` e `DB_BATCH_INTERVAL` controlam quantas leituras são agrupadas antes de gravar.
- **Detecção manual**: `isManualControl` permite distinguir comandos originados de outro painel – adapte se precisar travar o gráfico para modos específicos.
- **Reconexão WebSocket**: mude `WS_UPDATE_INTERVAL` ou os timers do heartbeat se o ambiente exigir TTL maior.

Troubleshooting

- **Botão “Iniciar” não faz nada**:
    - Veja o console → provavelmente faltou preencher um slider obrigatório.
    - Confirme se a serial está conectada; o backend recusa `/motion/start` sem porta aberta.
- **Gráfico vazio**:
    - Chart.js precisa estar carregado antes de `motion-logic.js`. Confira a ordem das tags `<script>`.
    - Verifique se o WebSocket recebeu dados (há logs no console quando `handleMotionTelemetry` roda).
- **CSV vazio**:
    - É necessário clicar em “Iniciar Gravação” antes de “Exportar CSV”.
    - IndexedDB pode estar bloqueado em ambientes privados; limpe os dados do site e tente novamente.
- **Status preso em “Parando...”**:
    - O ESP32 pode não ter respondido ao comando de stop. Abra o terminal do backend e veja se há exceções.
    - Use “Desconectar” na serial para forçar o stop e depois reconecte.

Autor

Guilherme Miyata - Instituto Federal de São Paulo (IFSP)
Trabalho de Conclusão de Curso - 2025

Github
Linkedin
Portfólio

Última atualização: Novembro 2025