

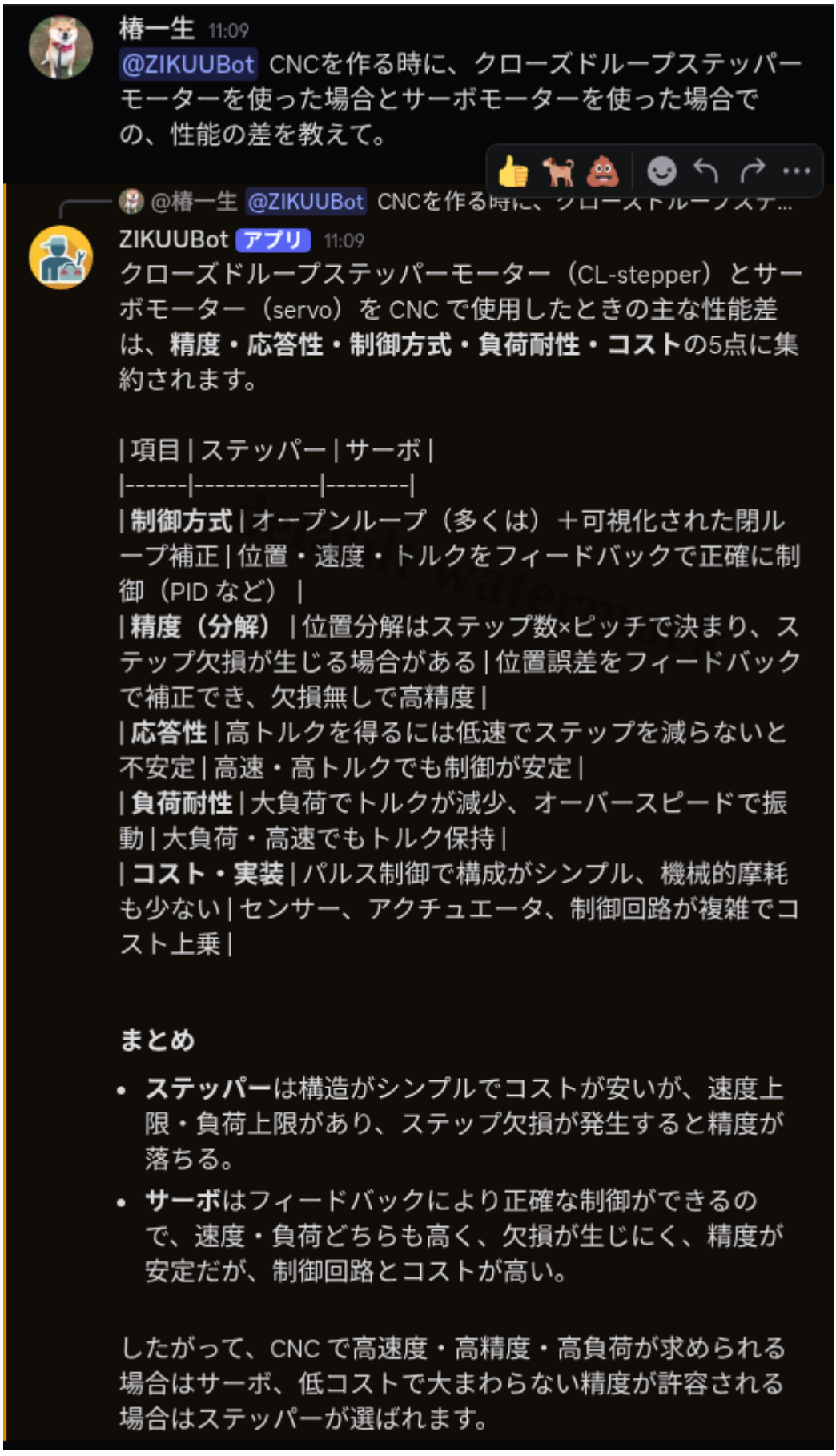
AIã?•ã?£ã??ã??ã??ã??ã??ã??LLM Brokerå¯¾å¿?ã•«ã•?ã??

Description

æ?¨æ?¥ã•«ç¶?ã•ã?•LLMã??ä½¿ç?¨ã•?ã•¦ã•?ã??**Discord**ã??ã??ã??ã??[LLM Broker](#)å¯¾å¿?ã•«ã•?ã•¦ã•?ã••ã•¾ã•?ã??

ä»?æ?¥ã•®ä¸?ç?ºç?®ã•¯AIã?•ã?£ã??ã??ã??ã??ã??ã??

default watermark



椿一生 11:09
@ZIKUUBot CNCを作る時に、クローズドループステッパーモーターを使った場合とサーボモーターを使った場合での、性能の差を教えて。

@椿一生 @ZIKUUBot CNCを作る時に、クローズドループステ...

ZIKUUBot アプリ 11:09
クローズドループステッパーモーター（CL-stepper）とサーボモーター（servo）を CNC で使用したときの主な性能差は、**精度・応答性・制御方式・負荷耐性・コスト**の5点に集約されます。

項目	ステッパー	サーボ
制御方式	オープンループ（多くは）＋可視化された閉ループ補正	位置・速度・トルクをフィードバックで正確に制御（PID など）
精度（分解）	位置分解はステップ数×ピッチで決まり、ステップ欠損が生じる場合がある	位置誤差をフィードバックで補正でき、欠損無しで高精度
応答性	高トルクを得るには低速でステップを減らないと不安定	高速・高トルクでも制御が安定
負荷耐性	大負荷でトルクが減少、オーバースピードで振動	大負荷・高速でもトルク保持
コスト・実装	パルス制御で構成がシンプル、機械的摩耗も少ない	センサー、アクチュエータ、制御回路が複雑でコスト上乗

まとめ

- **ステッパー**は構造がシンプルでコストが安いが、速度上限・負荷上限があり、ステップ欠損が発生すると精度が落ちる。
- **サーボ**はフィードバックにより正確な制御ができるので、速度・負荷どちらも高く、欠損が生じにくく、精度が安定だが、制御回路とコストが高い。

したがって、CNC で高速度・高精度・高負荷が求められる場合はサーボ、低コストで大まわらない精度が許容される場合はステッパーが選ばれます。

LLM Brokerå?¼ã•³å?ºã•?é?¨ã•¯ã?ã??ã•ªã?³ã?¼ã??ã•§ã•?ã??

LLM_PROFILEã•¨LLM_BROKER_URLã•¯ã?è¨®?ã??ã?¡ã?¤ã?«ã??docker-composeã??ã?¡ã?¤ã?«ã•?ã??èªã•¿è¾¼ã??æ?³å®?ã•§ã•?ã??

```
async def get_llm_response(messages_to_send):
    """Gets a response from LLM Broker."""
    payload = {
        "profile": LLM_PROFILE,
        "messages": messages_to_send,
    }

    try:
        async with httpx.AsyncClient(timeout=TIMEOUT) as client:
            response = await asyncio.wait_for(
                client.post(f"{LLM_BROKER_URL}/v1/chat", json=payload),
                timeout=TIMEOUT,
            )

        response.raise_for_status()
        data = response.json()

        meta = data.get("meta") or data.get("metadata") or {}
        if isinstance(meta, dict):
            logger.info(
                "llm_call_done: profile=%s backend=%s model=%s",
                data.get("profile") or meta.get("profile") or LLM_PROFILE,
                data.get("backend") or meta.get("backend"),
                data.get("model") or meta.get("model"),
            )
        else:
            logger.info("llm_call_done: profile=%s", LLM_PROFILE)

        return extract_broker_text(data).strip()

    except asyncio.TimeoutError:
        logger.warning("llm_call_timeout: profile=%s timeout=%.1f", LLM_PROFIL
        return "ã?¿ã?¤ã? ã?¢ã?¦ã??ã•?ã•¾ã•?ã•?ã??ã??ã•?ä¸?åº¦ã•?è©¦ã•?ã••ã•
 ã•?ã•?ã??"

    except httpx.HTTPStatusError as e:
        body = e.response.text[:500] if e.response is not None else ""
        logger.error(
            "llm_call_http_error: status=%s profile=%s body=%r",
            e.response.status_code if e.response is not None else None,
            LLM_PROFILE,
            body,
        )
        return f"LLM Brokerã•§ã?¨ã?©ã?¼ã•?ç?ºç??ã•?ã•¾ã•?ã•
?: HTTP {e.response.status_code}"

    except Exception as e:
        logger.exception("llm_call_error: profile=%s error=%s", LLM_PROFILE, e
        return f"ã?¨ã?©ã?¼ã•?ç?ºç??ã•?ã•¾ã•?ã•?: {e}"
```

default watermark

LLM_PROFILEã•¯ã?**LLM Broker**å•´ã•§ç®¡ç•?ã•¦ã•?ã??ã?•ã•?ã•®ã??ã?¡ã?¤ã?«ã•«è¨å®?ã•?ã??ã•¦ã•?ã??ã??ã•®ã•§ã?•AIã?•ã?£ã??ã??ã??ã??ã??ã•?ä½¿ã•£ã•¦ã•?ã??ã•®ã•?ã?•text_conversationã•¨ã•?ã•?ã??ã?ã??ã?£ã?¼ã?«ã•§ã•?ã??

```
# llm-broker/backend/profiles.yaml
profiles:
  text_extract:
    backend: llamacpp
    base_url: http://xxxxx
    model: gpt-oss-20b-Q4_K_M
    temperature: 0.1
    timeout_sec: 360

  text_conversation:
    backend: llamacpp
    base_url: http://xxxxx
    model: gpt-oss-20b-Q4_K_M
    temperature: 0.7
    timeout_sec: 360

  text_extract_ollama_fallback:
    backend: ollama
    base_url: http://xxxxx
    model: gpt-oss:20b
    temperature: 0.1
    timeout_sec: 360

  text_conversation_ollama_fallback:
    backend: ollama
    base_url: http://xxxxx
    model: gpt-oss:20b
    temperature: 0.7
    timeout_sec: 360
```

default watermark

Category

1. æ?¥ã? ã•®æ´»å??

Tags

1. AI
2. Discord
3. LLM
4. LLM Broker
5. ä¸?é??å??å¸?
6. å±±æ¢¨ç??

Date Created
2026å¹´7æ??7æ?¥
Author
kazuo-tsubaki