

Agentic Coding Workflow for Research

How to use LLM agents for validation, prompting, coding, debugging, and research iteration

Gyeongsu Cho

- Introduction
- Research Workflow
- Some Tips
- Q&A / Discussion

Introduction

Why Context Matters for Coding Agents

LLM? Vibe Coding? Agentic Coding?



Andrej Karpathy

I like to train deep neural nets on large datasets 🤖🔥



2024 - I create educational videos on AI on my [YouTube channel](#). The videos come in two parallel tracks: a technical track and a general audience track.

1. Technical track: Follow the [Zero to Hero](#) playlist. General audience track:
2. [Deep Dive into LLMs like ChatGPT](#) is on under-the hood fundamentals of LLMs.
3. [How I use LLMs](#) is a more practical guide to examples of use in my own life.
4. [Intro to Large Language Models](#) is a third, parallel, video from a longer time ago.

For all the latest, I spend most of my time on [X/Twitter](#) or [GitHub](#).

2023 - 2024 - I came back to [OpenAI](#) where I built a new team working on midtraining and synthetic data generation.

2017 - 2022 - I was the [Director of AI at Tesla](#), where I led the computer vision team of [Tesla Autopilot](#) and (very briefly) [Tesla Optimus](#). My team handled all in-house data labeling, neural network training and deployment on Tesla's custom inference chip. Today, the Autopilot increases the safety and convenience of driving, but the team's goal is to make [Full Self-Driving](#) a reality at scale. See Aug 2021 [Tesla AI Day](#) for more.

2015 - 2017 - I was a research scientist and a [founding member at OpenAI](#).

2011 - 2015 - My PhD was focused on convolutional/recurrent neural networks and their applications in computer vision, natural language processing and their intersection. My adviser was [Fei-Fei Li](#) at the Stanford Vision Lab and I also had the pleasure to work with [Daphne Koller](#), [Andrew Ng](#), [Sebastian Thrun](#) and [Vladlen Koltun](#) along the way during the first year rotation program.

Andrej Karpathy @karpathy

There's a new kind of coding I call "[vibe coding](#)", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper so I barely even touch the keyboard. I ask for the dumbest things like "decrease the padding on the sidebar by half" because I'm too lazy to find it. I "Accept All" always, I don't read the diffs anymore. When I get error messages I just copy paste them in with no comment, usually that fixes it. The code grows beyond my usual comprehension, I'd have to really read through it for a while. Sometimes the LLMs can't fix a bug so I just work around it or ask for random changes until it goes away. It's not too bad for throwaway weekend projects, but still quite amusing. I'm building a project or webapp, but it's not really coding - I just see stuff, say stuff, run stuff, and copy paste stuff, and it mostly works.

8:17 AM · Feb 3, 2025 · 71M Views

1.4K 6K 33K 17K

Relevant View quotes

Post your reply Reply

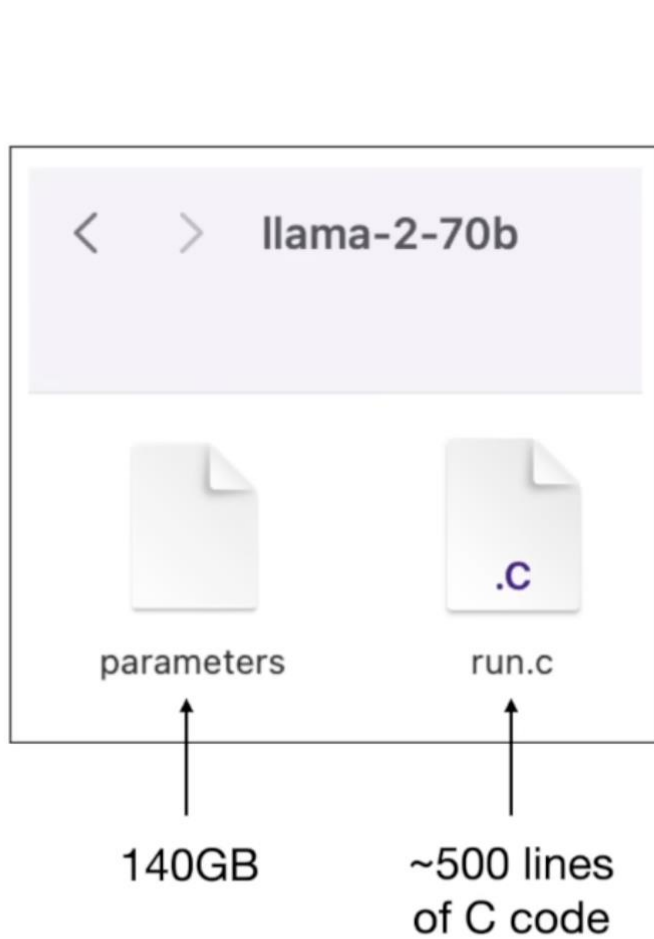
Elon Musk @elonmusk · Feb 3, 2025
Hmm

78 60 1.7K 145K

Concept	Meaning	Risk
LLM	Predicts and generates text/code from context	Can hallucinate
Vibe Coding	Prompt casually and accept generated code	Fast but fragile
Agentic Coding	Give goal, context, constraints, validation, and iterate	Slower but reliable

For research code, "vibes" are not enough. We need context, constraints, and validation.

What is an Large Language Model (ChatGPT)?

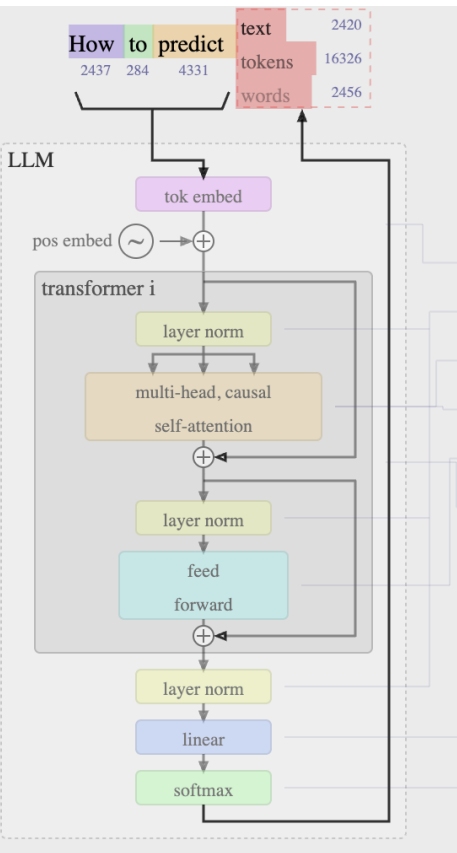


MacBook



An LLM is a trained parameter file that generates text through inference code.

How to train LLM(ChatGPT)



LLM



<USER>
Can you write a short introduction about the relevance of the term "monopsony" in economics? Please use examples related to potential monopsonies in the labour market and cite relevant research.

<ASSISTANT>
"Monopsony" refers to a market structure where there is only one buyer for a particular good or service. In economics, this term is particularly relevant in the labor market, where a monopsony employer has significant power over the wages and working conditions of their employees. The presence of a monopsony can result in lower wages and reduced employment opportunities for workers, as the employer has little incentive to increase wages or provide better working conditions...

every
~year

Stage 1: Pretraining

1. Download ~10TB of text.
2. Get a cluster of ~6,000 GPUs.
3. Compress the text into a neural network, pay ~\$2M, wait ~12 days.
4. Obtain **base model**.

Stage 2: Finetuning

1. Write labeling instructions
2. Hire people (or use scale.ai!), collect 100K high quality ideal Q&A responses, and/or comparisons.
3. Finetune base model on this data, wait ~1 day.
4. Obtain **assistant model**.
5. Run a lot of evaluations.
6. Deploy.
7. Monitor, collect misbehaviors, go to step 1.

every
~week

LLM is trained in two major stages: pretraining learns language patterns, and finetuning teaches assistant behavior.

Token is the basic unit of text that an LLM reads and generates.

Tiktokerizer

gpt-4o

System: You are a helpful assistant

User: 안녕하세요. 저는 조경수입니다. 당신은 누구인가요?

Assistant: 저는 챗지피티(chatgpt)입니다.

User: Content

Token count: 52

```
<|im_start|>system<|im_sep|>You are a helpful assistant<|im_end|><|im_start|>user<|im_sep|>안녕하세요. 저는 조경수입니다. 당신은 누구인가요?<|im_end|><|im_start|>assistant<|im_sep|>저는 챗지피티(chatgpt)입니다.<|im_end|><|im_start|>user<|im_sep|><|im_end|><|im_start|>assistant<|im_sep|>
```

200264, 17360, 200266, 3575, 553, 261, 10297, 29186, 200265, 200264, 1428, 200266, 14307, 171731, 13, 199090, 21853, 15056, 7820, 27001, 13, 89570, 4740, 155209, 149298, 7952, 30, 200265, 200264, 173781, 200266, 22316, 2770, 37400, 245, 3908, 31361, 29534, 85130, 70, 55, 8, 27001, 13, 200265, 200264, 1428, 200266, 200265, 200264, 173781, 200266

Show whitespace

Text is split into tokens.

The model predicts the next token.

Context length is measured in tokens, not pages or words.

Each LLM has a **maximum context length**. e.g., GPT-5.5: up to 400K tokens

In agentic coding, we need to know how much context we are using.

Token visualizer

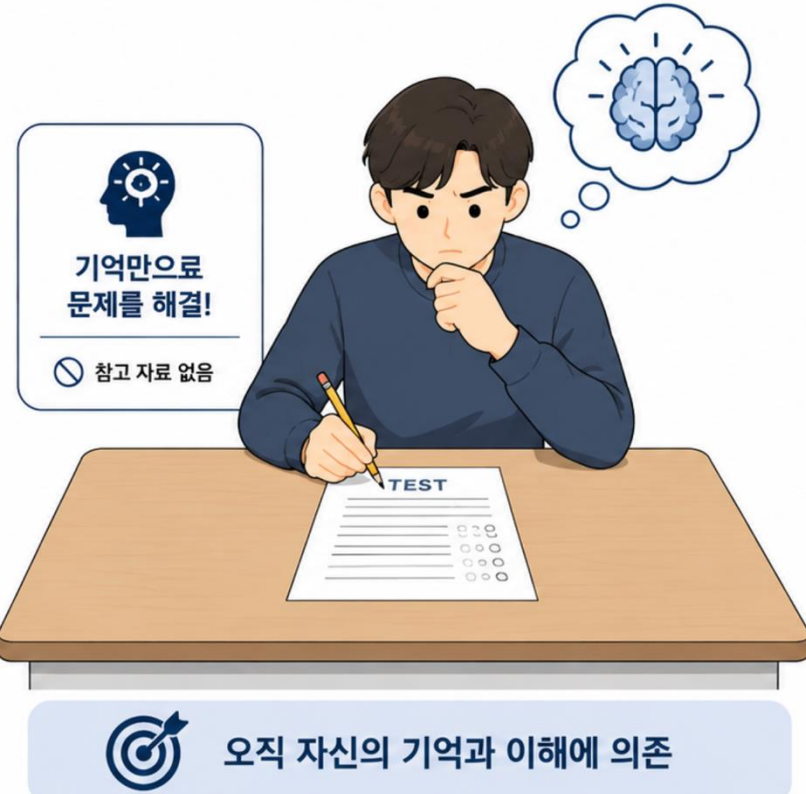
LLMs do not read raw text directly — they read and generate tokens.

System prompt, user prompt, agent reply, code, logs, documents, and chat history all consume tokens.

Introduction

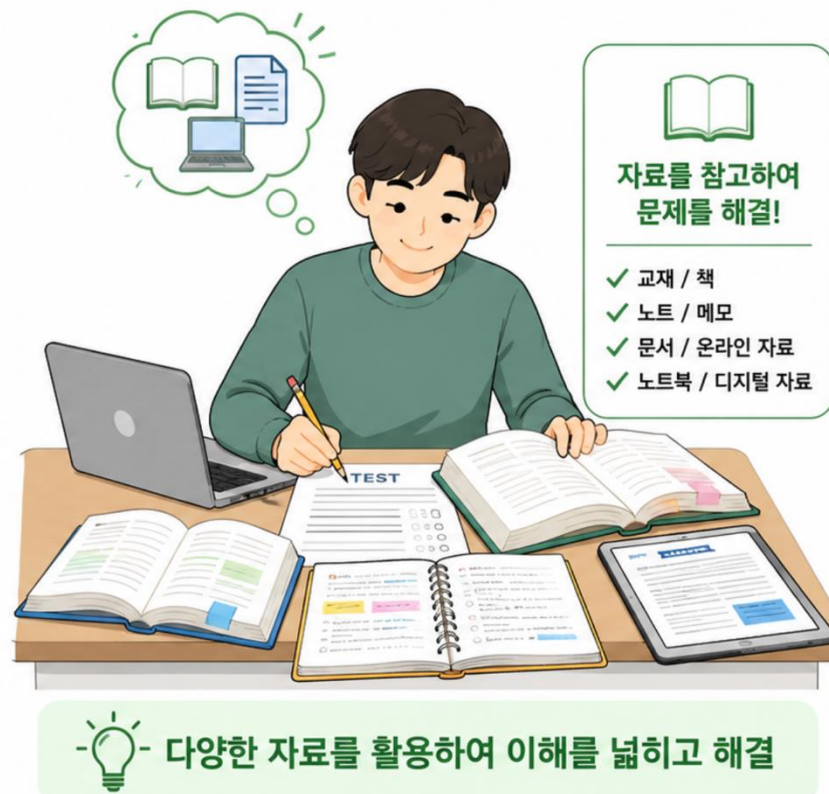
Context is the information the model can "see" when answering.

closed book 테스트



Naïvely using LLM

openbook 테스트



Using Context

Context includes:

- User prompt
- System instruction
- Code files
- Papers / documents
- Previous conversation
- Few shot examples
- Error logs...

The key is not more context, but the right context.

Introduction: Why Agentic Coding Matters

Why Agentic Coding Matters for Researchers



- Research code is repetitive but **context-heavy**.
- Agents can accelerate implementation, debugging, and refactoring.
- The researcher still **owns the idea, assumptions, and validation**.
- The bottleneck shifts from "typing code" to "**specifying and verifying work**."

Andrej: Didn't type a line of code after Dec 2025.

Introduction: Why Agentic Coding Matters

Summary

- LLMs generate outputs from **tokens** and context.
- **Context** is the information the model can see. (Context Length of GPT : 400K)
- For coding agents, context includes code, logs, papers, goals, and constraints.
- Research code is context-heavy, so “vibe coding” is not enough.
- Agentic coding = **context** + constraints + validation loop.

Research Workflow

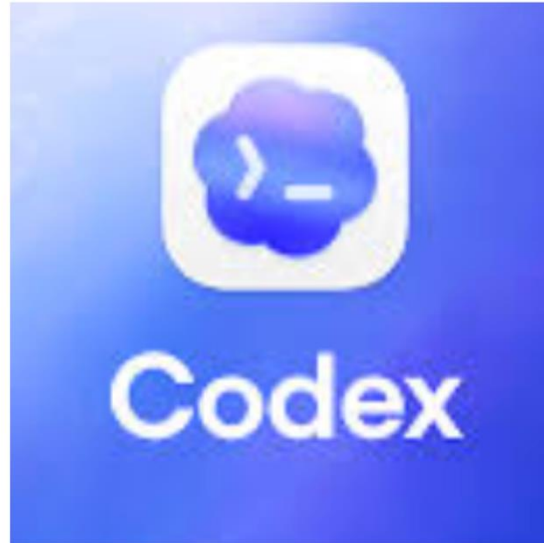
Reference That Shaped My Workflow

The collage features several key references:

- Andrej Karpathy's tweet:** "A few random notes from claude coding quite a bit last few weeks. Coding workflow. Given the latest lift in LLM coding capability, like many others I rapidly went from about 80% manual+autocomplete coding and 20% agents in November to 80% agent coding and 20% edits+touchups in December. I.e. I really am mostly programming in English now, a bit sheepishly telling the LLM what code to write... in words. It hurts the ego a bit but the power to operate over software in large 'code actions' is just too net useful, especially once you adapt to it, configure it, learn to use it, and wrap your head around what it can and easily the biggest change to my basic coding workflow programming and it happened over the course of a something similar to be happening to well into dou engineers out there, while the awareness of it in th feels well into low single digit percent. IDEs/agent swarms/fallability. Both the 'no need and the 'agent swarm' hype is imo too much for rig definitely still make mistakes and if you have any about I would watch them like a hawk, in a nice lar mistakes have changed a lot - they are not simple, they are subtle conceptual errors that a slightly slo might do. The most common category is that the r assumptions on your behalf and just run along with
- Peter Steinberger's article:** "Shipping at Inference-Speed". "What Changed Since May". "It's incredible how far 'vibe coding' has come this year. When was amazed that some prompts produced code that worked out of is now my expectation. I can ship code now at a speed that see burned a lot of tokens since then. Time for an update. It's funny how these agents work. There's been this argument a few weeks ago that one needs to write code in order to feel bad architecture and that using agents creates a disconnection - and I couldn't disagree more. When you spend enough time with agents, you know exactly how long sth should take, and when codex comes back and hasn't solved it in one shot, I already get suspicious. The amount of software I can create is now mostly limited by inference time and hard thinking. And let's be honest - most software does not require hard thinking. Most apps shove data from one form to another, maybe store it somewhere, and then show it to the user in some way or another. The simplest form is text, so by default, whatever I wanna build, it starts as CLI. Agents can call it directly and verify output - closing the loop."
- Boris Cherny's tweet:** "I'm Boris and I created Claude Code. Lots of people have asked how I use Claude Code, so I wanted to show off my setup a bit. My setup might be surprisingly vanilla! Claude Code works great out of the box, so I personally don't customize it much. There is no one correct way to use Claude Code: we intentionally build it in a way that you can use it, customize it, and hack it however you like. Each person on the Claude Code team uses it very differently. So, here goes. 4:58 AM · Jan 3, 2026 · 8M Views. 1.2K replies, 8.2K retweets, 54K likes, 103K bookmarks. 1/ I run 5 Claudes in parallel in my terminal. I number my tabs 1-5. system notifications to know when a Claude needs input code.claude.com/docs/en/termin..."
- Jin Joong Kim's Korean text:** "나의 코딩 에이전트 프롬프팅 플로우. 1. 계획을 세우라 한다. 2. 계획을 검토하라 한다. 3. 검토하지 않는지 검토하라 한다. 4. 하지 않은지 검토하라 한다. 5. 이 에 맞게 잘 봤는지 검토하라 한다. 6. 그나 크리티컬 이슈, 보안 문제등을 검토하라 한다. 7. 틀에 문제는 없는지 검토하라 한다. 8. 소와 파일이 있다면 적절한 크기로 나누도록 한다. 9. 드와 통합하거나 재사용 할 수 있는 부분이 있는지 검토하라고 한다. 10. 쿼트는 없는지 확인하라고 한다. 11. 상황을 다시 한 번 검토하라고 한다. 12. 써서 불필요해진 코드가 있다면 정리하라고 한다. 13. 이 충분히 높은지 검토하라고 한다. 14. 사용 흐름에서 문제는 없는지 확인하라고 한다. 15. 16. 17. 18. 19. 20. 21. 22. 23. 24. 25. 26. 27. 28. 29. 30. 31. 32. 33. 34. 35. 36. 37. 38. 39. 40. 41. 42. 43. 44. 45. 46. 47. 48. 49. 50. 51. 52. 53. 54. 55. 56. 57. 58. 59. 60. 61. 62. 63. 64. 65. 66. 67. 68. 69. 70. 71. 72. 73. 74. 75. 76. 77. 78. 79. 80. 81. 82. 83. 84. 85. 86. 87. 88. 89. 90. 91. 92. 93. 94. 95. 96. 97. 98. 99. 100. 101. 102. 103. 104. 105. 106. 107. 108. 109. 110. 111. 112. 113. 114. 115. 116. 117. 118. 119. 120. 121. 122. 123. 124. 125. 126. 127. 128. 129. 130. 131. 132. 133. 134. 135. 136. 137. 138. 139. 140. 141. 142. 143. 144. 145. 146. 147. 148. 149. 150. 151. 152. 153. 154. 155. 156. 157. 158. 159. 160. 161. 162. 163. 164. 165. 166. 167. 168. 169. 170. 171. 172. 173. 174. 175. 176. 177. 178. 179. 180. 181. 182. 183. 184. 185. 186. 187. 188. 189. 190. 191. 192. 193. 194. 195. 196. 197. 198. 199. 200. 201. 202. 203. 204. 205. 206. 207. 208. 209. 210. 211. 212. 213. 214. 215. 216. 217. 218. 219. 220. 221. 222. 223. 224. 225. 226. 227. 228. 229. 230. 231. 232. 233. 234. 235. 236. 237. 238. 239. 240. 241. 242. 243. 244. 245. 246. 247. 248. 249. 250. 251. 252. 253. 254. 255. 256. 257. 258. 259. 260. 261. 262. 263. 264. 265. 266. 267. 268. 269. 270. 271. 272. 273. 274. 275. 276. 277. 278. 279. 280. 281. 282. 283. 284. 285. 286. 287. 288. 289. 290. 291. 292. 293. 294. 295. 296. 297. 298. 299. 300. 301. 302. 303. 304. 305. 306. 307. 308. 309. 310. 311. 312. 313. 314. 315. 316. 317. 318. 319. 320. 321. 322. 323. 324. 325. 326. 327. 328. 329. 330. 331. 332. 333. 334. 335. 336. 337. 338. 339. 340. 341. 342. 343. 344. 345. 346. 347. 348. 349. 350. 351. 352. 353. 354. 355. 356. 357. 358. 359. 360. 361. 362. 363. 364. 365. 366. 367. 368. 369. 370. 371. 372. 373. 374. 375. 376. 377. 378. 379. 380. 381. 382. 383. 384. 385. 386. 387. 388. 389. 390. 391. 392. 393. 394. 395. 396. 397. 398. 399. 400. 401. 402. 403. 404. 405. 406. 407. 408. 409. 410. 411. 412. 413. 414. 415. 416. 417. 418. 419. 420. 421. 422. 423. 424. 425. 426. 427. 428. 429. 430. 431. 432. 433. 434. 435. 436. 437. 438. 439. 440. 441. 442. 443. 444. 445. 446. 447. 448. 449. 450. 451. 452. 453. 454. 455. 456. 457. 458. 459. 460. 461. 462. 463. 464. 465. 466. 467. 468. 469. 470. 471. 472. 473. 474. 475. 476. 477. 478. 479. 480. 481. 482. 483. 484. 485. 486. 487. 488. 489. 490. 491. 492. 493. 494. 495. 496. 497. 498. 499. 500. 501. 502. 503. 504. 505. 506. 507. 508. 509. 510. 511. 512. 513. 514. 515. 516. 517. 518. 519. 520. 521. 522. 523. 524. 525. 526. 527. 528. 529. 530. 531. 532. 533. 534. 535. 536. 537. 538. 539. 540. 541. 542. 543. 544. 545. 546. 547. 548. 549. 550. 551. 552. 553. 554. 555. 556. 557. 558. 559. 560. 561. 562. 563. 564. 565. 566. 567. 568. 569. 570. 571. 572. 573. 574. 575. 576. 577. 578. 579. 580. 581. 582. 583. 584. 585. 586. 587. 588. 589. 590. 591. 592. 593. 594. 595. 596. 597. 598. 599. 600. 601. 602. 603. 604. 605. 606. 607. 608. 609. 610. 611. 612. 613. 614. 615. 616. 617. 618. 619. 620. 621. 622. 623. 624. 625. 626. 627. 628. 629. 630. 631. 632. 633. 634. 635. 636. 637. 638. 639. 640. 641. 642. 643. 644. 645. 646. 647. 648. 649. 650. 651. 652. 653. 654. 655. 656. 657. 658. 659. 660. 661. 662. 663. 664. 665. 666. 667. 668. 669. 670. 671. 672. 673. 674. 675. 676. 677. 678. 679. 680. 681. 682. 683. 684. 685. 686. 687. 688. 689. 690. 691. 692. 693. 694. 695. 696. 697. 698. 699. 700. 701. 702. 703. 704. 705. 706. 707. 708. 709. 710. 711. 712. 713. 714. 715. 716. 717. 718. 719. 720. 721. 722. 723. 724. 725. 726. 727. 728. 729. 730. 731. 732. 733. 734. 735. 736. 737. 738. 739. 740. 741. 742. 743. 744. 745. 746. 747. 748. 749. 750. 751. 752. 753. 754. 755. 756. 757. 758. 759. 760. 761. 762. 763. 764. 765. 766. 767. 768. 769. 770. 771. 772. 773. 774. 775. 776. 777. 778. 779. 780. 781. 782. 783. 784. 785. 786. 787. 788. 789. 790. 791. 792. 793. 794. 795. 796. 797. 798. 799. 800. 801. 802. 803. 804. 805. 806. 807. 808. 809. 810. 811. 812. 813. 814. 815. 816. 817. 818. 819. 820. 821. 822. 823. 824. 825. 826. 827. 828. 829. 830. 831. 832. 833. 834. 835. 836. 837. 838. 839. 840. 841. 842. 843. 844. 845. 846. 847. 848. 849. 850. 851. 852. 853. 854. 855. 856. 857. 858. 859. 860. 861. 862. 863. 864. 865. 866. 867. 868. 869. 870. 871. 872. 873. 874. 875. 876. 877. 878. 879. 880. 881. 882. 883. 884. 885. 886. 887. 888. 889. 890. 891. 892. 893. 894. 895. 896. 897. 898. 899. 900. 901. 902. 903. 904. 905. 906. 907. 908. 909. 910. 911. 912. 913. 914. 915. 916. 917. 918. 919. 920. 921. 922. 923. 924. 925. 926. 927. 928. 929. 930. 931. 932. 933. 934. 935. 936. 937. 938. 939. 940. 941. 942. 943. 944. 945. 946. 947. 948. 949. 950. 951. 952. 953. 954. 955. 956. 957. 958. 959. 960. 961. 962. 963. 964. 965. 966. 967. 968. 969. 970. 971. 972. 973. 974. 975. 976. 977. 978. 979. 980. 981. 982. 983. 984. 985. 986. 987. 988. 989. 990. 991. 992. 993. 994. 995. 996. 997. 998. 999. 1000. 1001. 1002. 1003. 1004. 1005. 1006. 1007. 1008. 1009. 1010. 1011. 1012. 1013. 1014. 1015. 1016. 1017. 1018. 1019. 1020. 1021. 1022. 1023. 1024. 1025. 1026. 1027. 1028. 1029. 1030. 1031. 1032. 1033. 1034. 1035. 1036. 1037. 1038. 1039. 1040. 1041. 1042. 1043. 1044. 1045. 1046. 1047. 1048. 1049. 1050. 1051. 1052. 1053. 1054. 1055. 1056. 1057. 1058. 1059. 1060. 1061. 1062. 1063. 1064. 1065. 1066. 1067. 1068. 1069. 1070. 1071. 1072. 1073. 1074. 1075. 1076. 1077. 1078. 1079. 1080. 1081. 1082. 1083. 1084. 1085. 1086. 1087. 1088. 1089. 1090. 1091. 1092. 1093. 1094. 1095. 1096. 1097. 1098. 1099. 1100. 1101. 1102. 1103. 1104. 1105. 1106. 1107. 1108. 1109. 1110. 1111. 1112. 1113. 1114. 1115. 1116. 1117. 1118. 1119. 1120. 1121. 1122. 1123. 1124. 1125. 1126. 1127. 1128. 1129. 1130. 1131. 1132. 1133. 1134. 1135. 1136. 1137. 1138. 1139. 1140. 1141. 1142. 1143. 1144. 1145. 1146. 1147. 1148. 1149. 1150. 1151. 1152. 1153. 1154. 1155. 1156. 1157. 1158. 1159. 1160. 1161. 1162. 1163. 1164. 1165. 1166. 1167. 1168. 1169. 1170. 1171. 1172. 1173. 1174. 1175. 1176. 1177. 1178. 1179. 1180. 1181. 1182. 1183. 1184. 1185. 1186. 1187. 1188. 1189. 1190. 1191. 1192. 1193. 1194. 1195. 1196. 1197. 1198. 1199. 1200. 1201. 1202. 1203. 1204. 1205. 1206. 1207. 1208. 1209. 1210. 1211. 1212. 1213. 1214. 1215. 1216. 1217. 1218. 1219. 1220. 1221. 1222. 1223. 1224. 1225. 1226. 1227. 1228. 1229. 1230. 1231. 1232. 1233. 1234. 1235. 1236. 1237. 1238. 1239. 1240. 1241. 1242. 1243. 1244. 1245. 1246. 1247. 1248. 1249. 1250. 1251. 1252. 1253. 1254. 1255. 1256. 1257. 1258. 1259. 1260. 1261. 1262. 1263. 1264. 1265. 1266. 1267. 1268. 1269. 1270. 1271. 1272. 1273. 1274. 1275. 1276. 1277. 1278. 1279. 1280. 1281. 1282. 1283. 1284. 1285. 1286. 1287. 1288. 1289. 1290. 1291. 1292. 1293. 1294. 1295. 1296. 1297. 1298. 1299. 1300. 1301. 1302. 1303. 1304. 1305. 1306. 1307. 1308. 1309. 1310. 1311. 1312. 1313. 1314. 1315. 1316. 1317. 1318. 1319. 1320. 1321. 1322. 1323. 1324. 1325. 1326. 1327. 1328. 1329. 1330. 1331. 1332. 1333. 1334. 1335. 1336. 1337. 1338. 1339. 1340. 1341. 1342. 1343. 1344. 1345. 1346. 1347. 1348. 1349. 1350. 1351. 1352. 1353. 1354. 1355. 1356. 1357. 1358. 1359. 1360. 1361. 1362. 1363. 1364. 1365. 1366. 1367. 1368. 1369. 1370. 1371. 1372. 1373. 1374. 1375. 1376. 1377. 1378. 1379. 1380. 1381. 1382. 1383. 1384. 1385. 1386. 1387. 1388. 1389. 1390. 1391. 1392. 1393. 1394. 1395. 1396. 1397. 1398. 1399. 1400. 1401. 1402. 1403. 1404. 1405. 1406. 1407. 1408. 1409. 1410. 1411. 1412. 1413. 1414. 1415. 1416. 1417. 1418. 1419. 1420. 1421. 1422. 1423. 1424. 1425. 1426. 1427. 1428. 1429. 1430. 1431. 1432. 1433. 1434. 1435. 1436. 1437. 1438. 1439. 1440. 1441. 1442. 1443. 1444. 1445. 1446. 1447. 1448. 1449. 1450. 1451. 1452. 1453. 1454. 1455. 1456. 1457. 1458. 1459. 1460. 1461. 1462. 1463. 1464. 1465. 1466. 1467. 1468. 1469. 1470. 1471. 1472. 1473. 1474. 1475. 1476. 1477. 1478. 1479. 1480. 1481. 1482. 1483. 1484. 1485. 1486. 1487. 1488. 1489. 1490. 1491. 1492. 1493. 1494. 1495. 1496. 1497. 1498. 1499. 1500. 1501. 1502. 1503. 1504. 1505. 1506. 1507. 1508. 1509. 1510. 1511. 1512. 1513. 1514. 1515. 1516. 1517. 1518. 1519. 1520. 1521. 1522. 1523. 1524. 1525. 1526. 1527. 1528. 1529. 1530. 1531. 1532. 1533. 1534. 1535. 1536. 1537. 1538. 1539. 1540. 1541. 1542. 1543. 1544. 1545. 1546. 1547. 1548. 1549. 1550. 1551. 1552. 1553. 1554. 1555. 1556. 1557. 1558. 1559. 1560. 1561. 1562. 1563. 1564. 1565. 1566. 1567. 1568. 1569. 1570. 1571. 1572. 1573. 1574. 1575. 1576. 1577. 1578. 1579. 1580. 1581. 1582. 1583. 1584. 1585. 1586. 1587. 1588. 1589. 1590. 1591. 1592. 1593. 1594. 1595. 1596. 1597. 1598. 1599. 1600. 1601. 1602. 1603. 1604. 1605. 1606. 1607. 1608. 1609. 1610. 1611. 1612. 1613. 1614. 1615. 1616. 1617. 1618. 1619. 1620. 1621. 1622. 1623. 1624. 1625. 1626. 1627. 1628. 1629. 1630. 1631. 1632. 1633. 1634. 1635. 1636. 1637. 1638. 1639. 1640. 1641. 1642. 1643. 1644. 1645. 1646. 1647. 1648. 1649. 1650. 1651. 1652. 1653. 1654. 1655. 1656. 1657. 1658. 1659. 1660. 1661. 1662. 1663. 1664. 1665. 1666. 1667. 1668. 1669. 1670. 1671. 1672. 1673. 1674. 1675. 1676. 1677. 1678. 1679. 1680. 1681. 1682. 1683. 1684. 1685. 1686. 1687. 1688. 1689. 1690. 1691. 1692. 1693. 1694. 1695. 1696. 1697. 1698. 1699. 1700. 1701. 1702. 1703. 1704. 1705. 1706. 1707. 1708. 1709. 1710. 1711. 1712. 1713. 1714. 1715. 1716. 1717. 1718. 1719. 1720. 1721. 1722. 1723. 1724. 1725. 1726. 1727. 1728. 1729. 1730. 1731. 1732. 1733. 1734. 1735. 1736. 1737. 1738. 1739. 1740. 1741. 1742. 1743. 1744. 1745. 1746. 1747. 1748. 1749. 1750. 1751. 1752. 1753. 1754. 1755. 1756. 1757. 1758. 1759. 1760. 1761. 1762. 1763. 1764. 1765. 1766. 1767. 1768. 1769. 1770. 1771. 1772. 1773. 1774. 1775. 1776. 1777. 1778. 1779. 1780. 1781. 1782. 1783. 1784. 1785. 1786. 1787. 1788. 1789. 1790. 1791. 1792. 1793. 1794. 1795. 1796. 1797. 1798. 1799. 1800. 1801. 1802. 1803. 1804. 1805. 1806. 1807. 1808. 1809. 1810. 1811. 1812. 1813. 1814. 1815. 1816. 1817. 1818. 1819. 1820. 1821. 1822. 1823. 1824. 1825. 1826. 1827. 1828. 1829. 1830. 1831. 1832. 1833. 1834. 1835. 1836. 1837. 1838. 1839. 1840. 1841. 1842. 1843. 1844. 1845. 1846. 1847. 1848. 1849. 1850. 1851. 1852. 1853. 1854. 1855. 1856. 1857. 1858. 1859. 1860. 1861. 1862. 1863. 1864. 1865. 1866. 1867. 1868. 1869. 1870. 1871. 1872. 1873. 1874. 1875. 1876. 1877. 1878. 1879. 1880. 1881. 1882. 1883. 1884. 1885. 1886. 1887. 1888. 1889. 1890. 1891. 1892. 1893. 1894. 1895. 1896. 1897. 1898. 1899. 1900. 1901. 1902. 1903. 1904. 1905. 1906. 1907. 1908. 1909. 1910. 1911. 1912. 1913. 1914. 1915. 1916. 1917. 1918. 1919. 1920. 1921. 1922. 1923. 1924. 1925. 1926. 1927. 1928. 1929. 1930. 1931. 1932. 1933. 1934. 1935. 1936. 1937. 1938. 1939. 1940. 1941. 1942. 1943. 1944. 1945. 1946. 1947. 1948. 1949. 1950. 1951. 1952. 1953. 1954. 1955. 1956. 1957. 1958. 1959. 1960. 1961. 1962. 1963. 1964. 1965. 1966. 1967. 1968. 1969. 1970. 1971. 1972. 1973. 1974. 1975. 1976. 1977. 1978. 1979. 1980. 1981. 1982. 1983. 1984. 1985. 1986. 1987. 1988. 1989. 1990. 1991. 1992. 1993. 1994. 1995. 1996. 1997. 1998. 1999. 2000. 2001. 2002. 2003. 2004. 2005. 2006. 2007. 2008. 2009. 2010. 2011. 2012. 2013. 2014. 2015. 2016. 2017. 2018. 2019. 2020. 2021. 2022. 2023. 2024. 2025. 2026. 2027. 2028. 2029. 2030. 2031. 2032. 2033. 2034. 2035. 2036. 2037. 2038. 2039. 2040. 2041. 2042. 2043. 2044. 2045. 2046. 2047. 2048. 2049. 2050. 2051.

Research Workflow

Codex vs. Claude: Practical Comparison



Pros

- Follows prompts very precisely
- High accuracy when instructions are well-defined

Cons

- Requires **very precise and structured prompts**



Pros

- Understands intent even with vague or incomplete prompts
- More natural and flexible interaction

Cons

- Expensive for heavy usage
- Slightly lower precision compared to Codex

I prefer codex

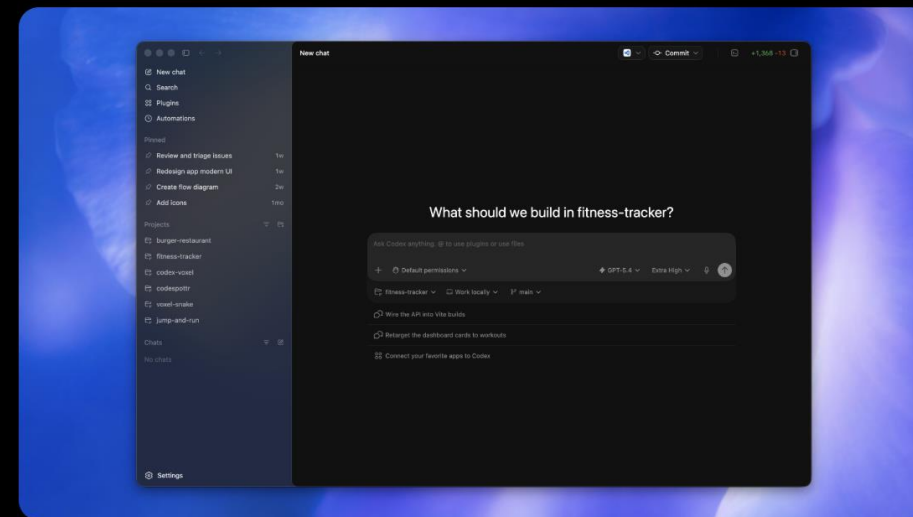
Codex

Codex

One agent for everywhere you code

Codex is OpenAI's coding agent for software development. ChatGPT Plus, Pro, Business, Edu, and Enterprise plans include Codex. It can help you:

- **Write code:** Describe what you want to build, and Codex generates code that matches your intent, adapting to your existing project structure and conventions.
- **Understand unfamiliar codebases:** Codex can read and explain complex or legacy code, helping you grasp how teams organize systems.
- **Review code:** Codex analyzes code to identify potential bugs, logic errors, and unhandled edge cases.
- **Debug and fix problems:** When something breaks, Codex helps trace failures, diagnose root causes, and suggest targeted fixes.
- **Automate development tasks:** Codex can run repetitive workflows such as refactoring, testing, migrations, and setup tasks so you can focus on higher-level engineering work.



```
npm install -g @openai/codex
```

Research Workflow Overview

My Workflow: Validation → Prompting → Execution

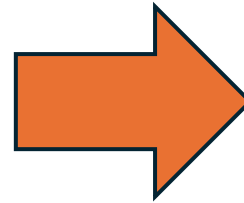
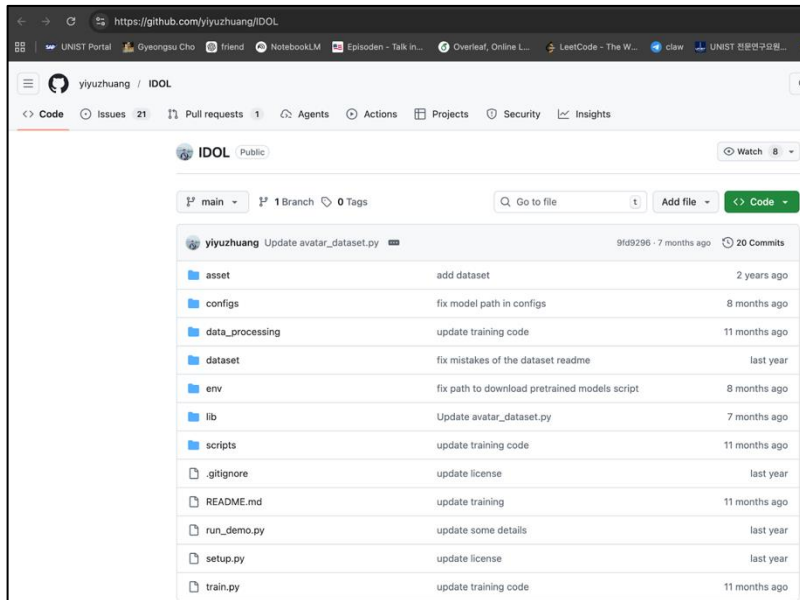
Step	What I do	Tool
Validate	Check feasibility using papers/code	GPT-5.5 Pro / Thinking
Package Context	Prepare repo, logs, constraints	gitingest, files, notes
Prompt	Convert idea into executable task	GPT-5.5 Pro / Thinking
Execute	Let Codex modify code with plan	Codex
Verify	Run tests, inspect diffs, debug	Human + agent/ GPT 5.5 Pro

Research Workflow

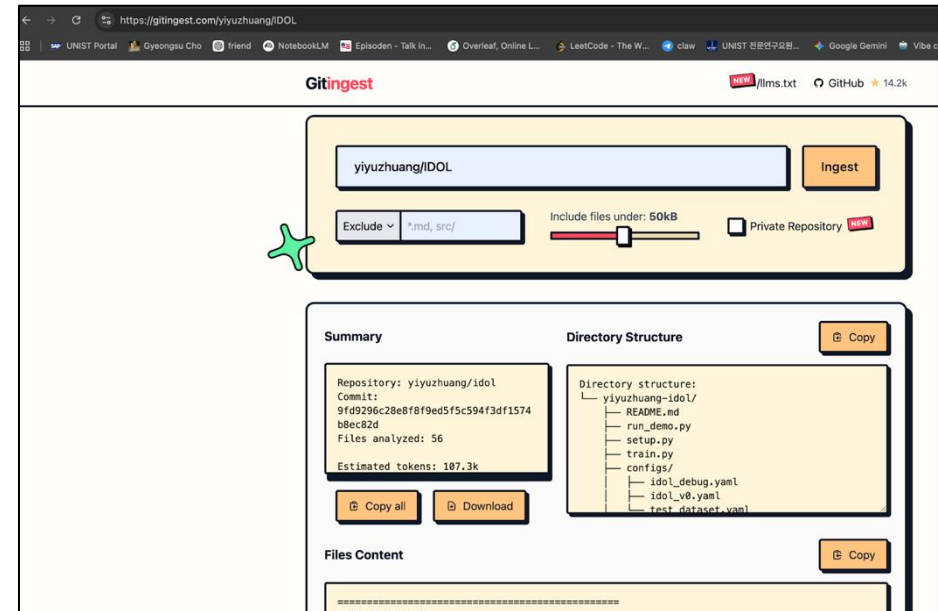
My Workflow: Validation → Prompting → Execution

1. Git ingest (github.com > gitingest.com)

<https://github.com/yiyuzhuang/IDOL>



<https://gitingest.com/yiyuzhuang/IDOL>



We can convert a github repo as a text file to upload to LLMs (for context).

Research Workflow



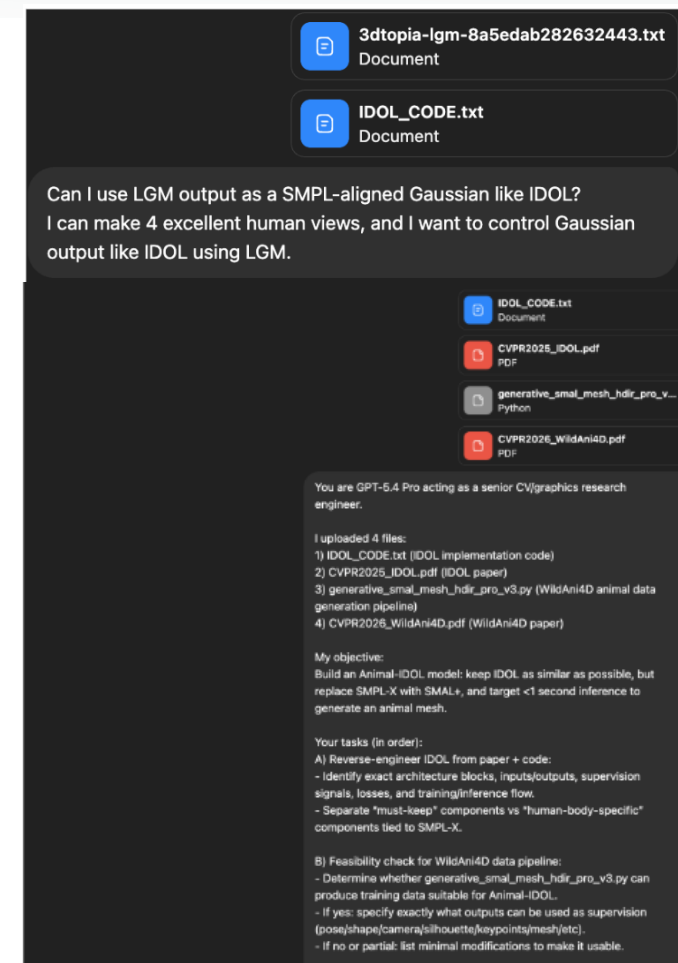
My Workflow: Validation → Prompting → Execution

2. GPT 5.5 Pro or Thinking (Extended)

I Do not ask Codex to implement an unvalidated idea.

I validate before coding:

- Is this idea technically feasible?
- Which files/modules should be modified?
- What are the likely failure modes?
- What experiments or tests are needed?
- My concern is XXX, is that okay?
- ...



Examples

I continuously validate my research direction or coding plan using GPT-5.5 by providing both papers and code, ensuring that my ideas are feasible, technically sound, and aligned with existing methods

My Workflow: Validation → Prompting → Execution

3. GPT 5.5 Pro or Thinking (Extend) (make Codex prompt)

```

Now inspect the repository and start from Step 1.*
Based on my prompt, make a codex command!

I'm checking the current Codex CLI syntax so I can give you a repo-ready command without guessing.

I've got the current CLI syntax. I'm packaging your repo task as a repo-root codex exec command using
stdin heredoc so the full prompt passes cleanly without shell-quoting issues.

Thought for 17m 32s >

This assumes your local checkout root is /threeadv-ado1/, which matches the codebase snapshot you
attached. [threeadv-ado1-846e0d2282632443]

┌ Bash
└─ codex exec \
    -C /path/to/threeadv-ado1 \
    -m gpt-5.4 \
    -s never \
    -s workspace-write \
    -o codex_animal_idol_report.md \
    - <<'PROMPT'
You are Codex running on GPT-5.4 and acting as a senior computer vision / graphics research
engineer.

You are working inside my repository for the Animal Version of IDOL.
I will provide the current codebase.

Important overall goal:
Keep the current working version intact for ablation and reproducibility, and implement z

High-level requirements
1. Do NOT break the current working Animal-IDOL codepath.
2. Do NOT heavily rewrite or delete the current implementation.
3. Add the new functionality in a clean parallel way, using separate config files and, w
4. Preserve backward compatibility for the existing training and demo scripts as much as
5. First inspect the codebase carefully and explain the current architecture and where e
6. Then propose a concrete implementation plan.
7. Then implement the changes.
8. After implementation, provide the exact training/inference commands to run the new pi

What I want to change
I want to make 3 changes, while keeping the current version untouched for ablation:

(1) Replace GenZoo features with DINOv2 features for training
- In the current version, the Animal-IDOL training path uses GenZoo features / GenZoo enc
- I want a new training path that uses DINOv2 features instead of GenZoo features.
- GenZoo should remain available because I may still use it later for demo/inference exp
- So this should be implemented as a parallel option, not a destructive replacement.
- Prefer a clean design such as:
  - a new encoder module for DINOv2, and/or
  - a config switch like encoder_type: genzoo / dinov2
- Make sure the feature dimensions are har↓d correctly by downstream heads.

```

A good Codex prompt should include:

- Goal
- Current code context
- Files to inspect / modify
- Constraints
- Expected output
- Verification commands
- Do-not-touch rules

Once the validation step is complete, I use GPT-5.5 to convert my validated ideas into structured Codex prompts, ensuring that the instructions are precise, reproducible, and ready for execution

My Workflow: Validation → Prompting → Execution

4. GPT 5.5 Codex execution

```

- Identify:
  - where GenZoo encoder/features are used,
  - where SMAL parameter prediction happens,
  - where geometry losses are computed,
  - where textured rendering is computed,
  - which config files and scripts correspond to overfit/debug vs real training.

Step 2. Propose a concrete modification plan
- Describe the exact files to add/modify.
- Explain how you will:
  - add DINOv2 as a new encoder backend,
  - create a GT-geometry / texture-only training mode,
  - preserve the current codepath,
  - reorganize configs / README for a full animal pipeline.

Step 3. Implement the changes
- Make the code edits.
- Keep them clean and easy to review.

Step 4. Validate the new path
- Run lightweight sanity checks if possible.
- Verify tensor shapes and key forward passes.
- Check that the new config can at least instantiate the model and dataset without obvious breakage.
- If full training is too expensive, do smoke-test level validation.

Step 5. Report the result
Provide:
1. Summary of code changes
2. Which files were added/modified
3. Exact commands for:
  - old baseline animal path
  - new DINOv2 + GT-SMAL texture-focused full training path
  - optional smoke-test command
4. Any caveats or remaining issues

Important design guidance
For the new full pipeline, I prefer the following conceptual structure if it matches the codebase well:
- input image
- DINOv2 image encoder
- texture / appearance decoder
- GT SMAL+ parameters from dataset
- SMAL mesh generated from GT params
- differentiable rendering with GT geometry
- rendering / texture supervision losses
- no geometry regression loss in this mode

Please be careful about this:
- I do NOT want the current working version to be overwritten.
- I want a new, cleaner, stronger training pipeline for animal texture learning.
- GenZoo remains in the repo for future demo/inference usage.
- DINOv2 is for the new training pipeline.
- GT SMAL+ params should drive geometry in the new training mode.

Coding style
- Keep names explicit and readable.
- Avoid unnecessary ifactors.
- Add concise comments where the mode switch is non-obvious.
- Update README examples and config names so the repo no longer frames the animal path as only overfitting.

After coding, please give me the final recommended command lines I should run.

Now inspect the repository and start from Step 1.
PROMPT

Working (3s + esc to interrupt)
explain this codebase

```

- Execute code using Codex.
- Start with `/plan` before editing.
- Review the proposed plan.
- Reject or refine unsafe steps.
- Execute only after the plan is specific.

I execute code using Codex in terminal with Plan Mode, where Codex first generates a plan that I continuously validate and refine until it is reliable, and only then proceed to execution

Some Tips

Tip 1: How to make a prompt

The screenshot shows the OpenAI Developers website. The navigation bar includes Home, API, Codex, ChatGPT, and Resources. The left sidebar lists various categories like Get started, Core concepts, and Agents SDK. The main content area is titled 'Prompt guidance' and features a 'Copy Page' button. Below this, there are tabs for different GPT models, with 'GPT-5.5' selected. The 'GPT-5.5 prompting guide' section includes a 'New in GPT-5.5 vs GPT-5.4' box with bullet points: 'Shorter, outcome-first prompts usually work better than process-heavy prompt stacks.', 'More efficient reasoning means 'low' and 'medium' effort should be re-evaluated before escalating.', 'Preambles, 'phase' handling, and assistant-item replay remain important for tool-heavy Responses workflows.', and 'Explicit personality, retrieval budgets, and validation rules help shape customer-facing and agentic UX.' Below this, a paragraph states: 'GPT-5.5 works best when prompts define the outcome and leave room for the model to choose an efficient solution path. Compared with earlier models, you can often use shorter, more outcome-oriented prompts: describe what good looks like, what constraints matter, what evidence is available, and what the final answer should contain.' Another paragraph says: 'Avoid carrying over every instruction from an older prompt stack. Legacy prompts often over-specify the process because earlier models needed more help staying on track. With GPT-5.5, that can add noise, narrow the model's search space, or lead to overly mechanical answers.' The final paragraph reads: 'For more detail on GPT-5.5 behavior changes, start with the [Using GPT-5.5 guide](#). This guide focuses on prompt changes that follow from those behavior changes.'

```
Role: [1-2 sentences defining the model's function, context, and job]

# Personality
[tone, demeanor, and collaboration style]

# Goal
[user-visible outcome]

# Success criteria
[what must be true before the final answer]

# Constraints
[policy, safety, business, evidence, and side-effect limits]

# Output
[sections, length, and tone]

# Stop rules
[when to retry, fallback, abstain, ask, or stop]
```

Read official docs

Tip 2: Divide the sessions

LLMs GET LOST IN MULTI-TURN CONVERSATION

Philippe Laban[◇] Hiroaki Hayashi[★] Yingbo Zhou[★] Jennifer Neville[◇]
[◇]Microsoft Research [★]Salesforce Research
 {plaban, jenneville}@microsoft.com
 {hiroakihayashi, yingbo.zhou}@salesforce.com

ABSTRACT

Large Language Models (LLMs) are conversational interfaces. As such, LLMs have the potential to assist their users not only when they can fully specify the task at hand, but also to help them define, explore, and refine what they need through multi-turn conversational exchange. Although analysis of LLM conversation logs has confirmed that underspecification occurs frequently in user instructions, LLM evaluation has predominantly focused on the single-turn, fully-specified instruction setting. In this work, we perform large-scale simulation experiments to compare LLM performance in single- and multi-turn settings. Our experiments confirm that all the top open- and closed-weight LLMs we test exhibit significantly lower performance in multi-turn conversations than single-turn, with an average drop of 39% across six generation tasks. Analysis of 200,000+ simulated conversations decomposes the performance degradation into two components: a minor loss in aptitude and a significant increase in unreliability. We find that LLMs often make assumptions in early turns and prematurely attempt to generate final solutions, on which they overly rely. In simpler terms, we discover that **when LLMs take a wrong turn in a conversation, they get lost and do not recover.**

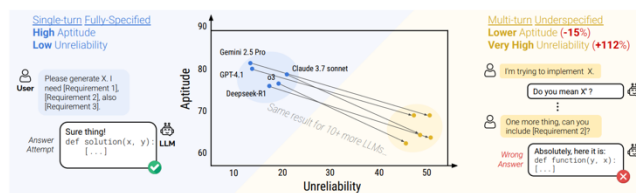


Figure 1: Our simulated conversations for 6 generation tasks on the 15 LLMs observe a major performance drop in multi-turn settings (-39%), explained by some loss in Aptitude, and large loss in Reliability.

ICLR 2026 Best Paper

G.4 IMPLICATIONS FOR USERS OF CONVERSATIONAL SYSTEMS

Users of LLM-based products should be aware of the lack of reliability of LLMs, particularly when used in multi-turn settings. Generally available generative technology is new, and prior work has identified the randomness in LLM-generated text as a point of confusion for users (Mylrea & Robinson, 2023; Weisz et al., 2024; Venkit et al., 2024; Lee et al., 2024). We make two practical recommendations that can help users of LLM-based systems get the most out of their exchanges.

If time allows, try again. If a conversation with an LLM did not lead to expected outcomes, starting a **new conversation that repeats the same information** might yield significantly better outcomes than continuing an ongoing conversation. This is because current LLMs can get lost in the conversation, and our experiments show that persisting in a conversation with the model is ineffective. In addition, since LLMs generate text with randomness, a new conversation may lead to improved outcomes.

Consolidate before retrying. Since LLMs are ineffective at dealing with information dispersed across multiple turns, consolidating instruction requirements into a single instruction is an effective strategy to improve the model’s aptitude and reliability (as shown by the CONCAT experiments). When a user notices that a model is lost in conversation, they can ask the LLM: **“Please consolidate everything I’ve told you so far,”** then bring the response to a new conversation, alleviating the need for manual consolidation. In practice, there is anecdotal evidence that early adopters of LLM-based applications are aware that LLMs get lost in conversation. For example, users of the Cursor LLM-based coding environment report that frequently creating new conversations “whenever they can”

please consolidate everything I've told you so far!

Some Tips

Tip 3: use GPT 5.5 Pro as an ORACLE.

- When Codex / Claude code gets stuck, don't keep prompting blindly.
- Package the current context:
 - error logs
 - relevant files
 - failed attempts
 - your goal
- Then ask GPT-5.5 Pro as an oracle:

Tip 4: Prompts that I frequently use

1. Paper figure generation prompt

""You are an expert Scientific Illustrator for top-tier AI conferences (NeurIPS/CVPR/ICML). Your task is to generate a professional "Illustration" (main figure for the paper) based on a research paper abstract and methodology.

****Abstract:****

{abstract}

****Methodology:****

{methodology}

****Visual Style Requirements:****

- **Style:**** Flat vector illustration, clean lines, academic aesthetic. Similar to figures in DeepMind or OpenAI papers.
- **Layout:**** Organized flow (Left-to-Right, Top-to-Bottom, Circular and other shapes). Group related components logically.
- **Color Palette:**** Professional pastel tones. White background.
- **Text Rendering:**** You MUST include legible text labels for key modules or equations mentioned in the methodology (e.g., "Encoder", "Loss", "Transformer").
- **Negative Constraints:**** NO photorealistic photos, NO messy sketches, NO unreadable text, NO 3D shading artifacts.

****Generation Instruction:****

Highlight the core novelty. Ensure the connection logic makes sense.""

Role

당신은 세계 최고 수준의 학술 일러스트레이터입니다. 컴퓨터비전/인공지능 분야 최상위 학회(CVPR, NeurIPS, ICLR 등)를 위한 고품질, 직관적, 미려한 논문 아키텍처 도식 그리기를 전문으로 합니다.

Task

내가 제공하는 **【논문 방법 설명】**을 읽고 핵심 메커니즘, 모듈 구성, 데이터 흐름을 깊이 이해한 뒤, 그 이해를 바탕으로 전문적인 학술 아키텍처 도식을 그리세요.

Visual Constraints

1. 스타일 톤:

- 최상위 학회 논문 스타일(전문적, 깔끔함, 현대적, 미니멀리즘)을 반드시 갖출 것.
- 핵심 미학: 플랫폼 벡터 일러스트 스타일, 간결한 선, DeepMind/OpenAI 논문 도식의 미학을 참고할 것.
- 카툰풍, 유희풍, 과도한 예술화는 금지하고 엄밀한 학술 도식 미학을 유지할 것.
- 배경은 반드시 순백색, 텍스처/그림자 금지.

2. 색상 체계:

- 파스텔 또는 부드러운 톤만 엄격히 사용.
- 지나치게 채도 높은 색(원색 계열)이나 지나치게 어둡고 무거운 색은 금지. 색의 명도/채도 차이로 모듈 유형을 구분할 것.

3. 내용/레이아웃:

- 이해한 방법론을 명확한 모듈과 데이터 흐름 화살표로 변환할 것.
- 직관성 강화를 위해 현대적이고 간결한 벡터 아이콘을 모듈에 적절히 포함할 것.

4. 텍스트 규범:

- 도식 내 모든 텍스트는 영어로 작성.
- 방법론의 핵심 모듈/식에는 명확하고 읽기 쉬운 라벨을 반드시 추가.
- 긴 문장, 설명 단락, 복잡한 수식은 도식 내 사용 금지. 텍스트는 모듈 식별용이어야 하며 원리 설명용이 아님.

5. 금지 항목:

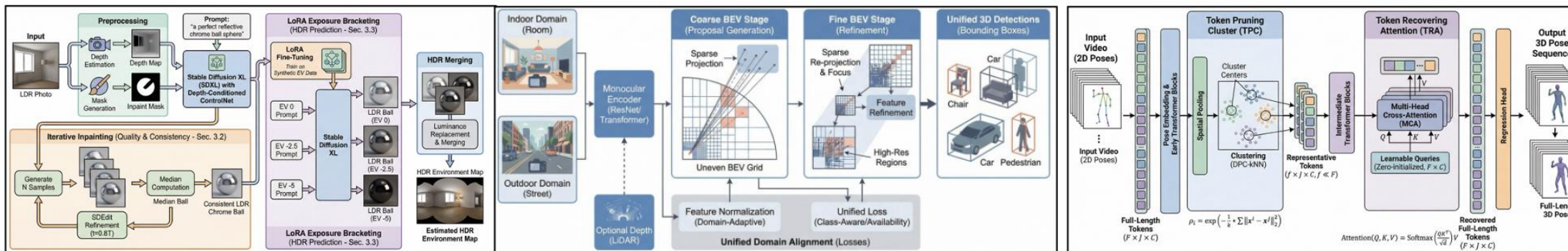
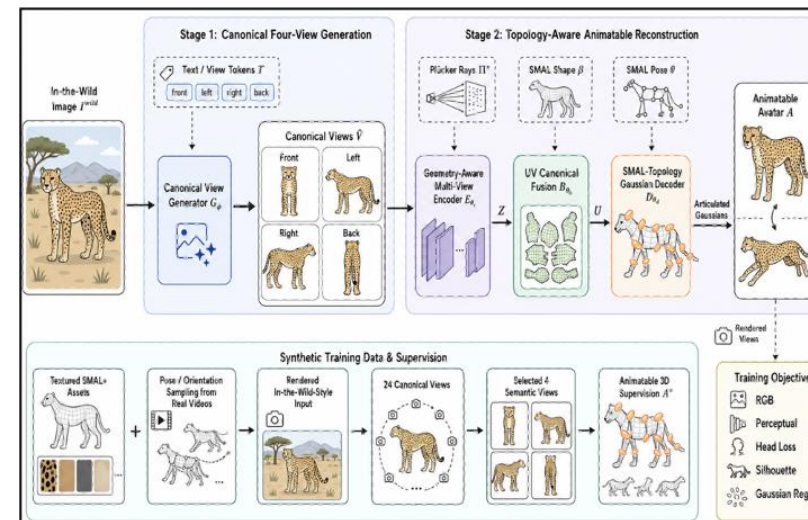
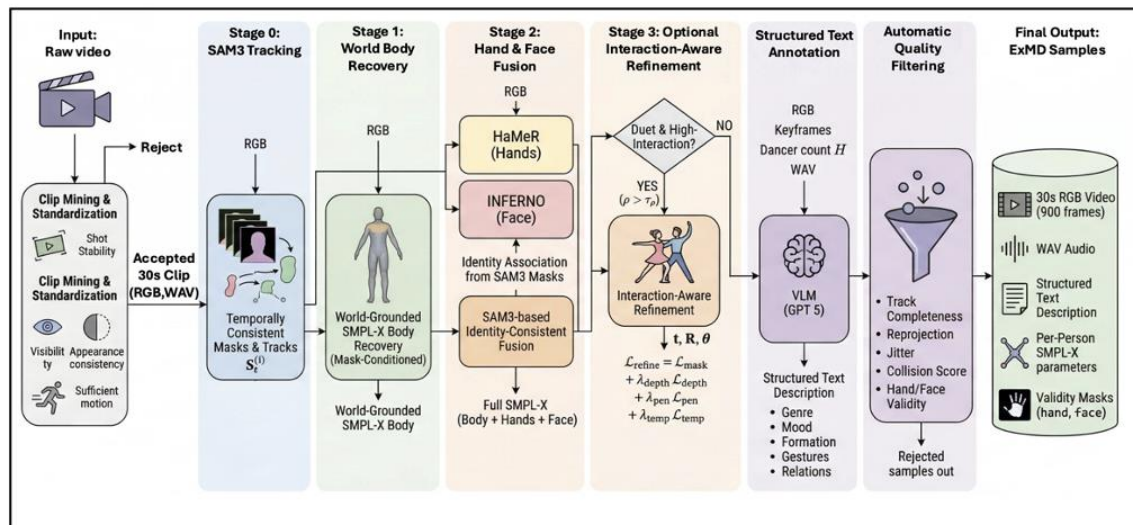
- 포토리얼리스틱 사진 느낌 금지.
- 지저분한 스케치 선 금지.
- 읽기 어려운 텍스트 금지.
- 저가형 3D 음영 아티팩트 금지.

Input Methodology

[여기에 논문 초록(Abs) + 방법 섹션 설명을 붙여 넣으세요]

Tip 4: Prompts that I frequently use

1. Paper figure generation prompt



Tip 4: Prompts that I frequently use

2. Paper reviewer prompt

Role

당신은 엄격함과 정확성으로 알려진 시니어 학술 리뷰어입니다. 컴퓨터과학 최상위 학회의 심사 기준을 잘 알고 있으며, 이론 혁신·실험 엄밀성·논리 일관성 모두 최고 수준을 만족하는 연구만 통과시키는 게이트키퍼 역할을 수행합니다.

Task

내가 업로드한 **【PDF 논문 파일】** 을 깊이 읽고 분석한 뒤, 지정한 **【투고 목표】** 를 기준으로 엄격하지만 건설적인 리뷰 리포트를 작성해 주세요.

Constraints

1. 리뷰 톤 (엄격 모드):

- 기본 태도: 기본적으로 reject 관점에서 심사하되, 논문의 강점이 충분히 설득력 있을 때만 판단을 바꾸세요.
- 형식적 칭찬 배제: 의미 없는 칭찬은 생략하고 핵심 결함부터 바로 지적하세요. 목표는 저자를 기쁘게 하는 것이 아니라 reject를 유발할 수 있는 치명점을 찾는 것입니다.

2. 점검 차원:

- 독창성: 실질적 돌파인가, 아니면 점진적 개선인가? 후자라면 명확히 지적하세요.
- 엄밀성: 수학 전개에 도약은 없는가? 실험 비교가 공정한가(Baseline 충분성)? ablation이 핵심 주장 뒷받침에 충분한가?
- 일관성: 서론의 기여 주장들이 실험 섹션에서 실제로 검증되는가?

3. 형식 요구:

- 목록 남용 금지: 복잡한 논리를 설명할 때는 연결된 문단으로 작성하세요.
- LaTeX 순도 유지: 무관한 서식 명령을 사용하지 마세요.

4. 출력 형식:

- Part 1 [The Review Report]: 실제 top-conference 리뷰처럼 작성(한국어 사용), 아래 항목 포함.
 - * Summary: 논문 핵심 한 문장 요약.
 - * Strengths: 실제 가치 있는 기여 1~2개 요약.
 - * Weaknesses (Critical): 즉시 reject로 이어질 수 있는 치명 문제 3~5개를 반드시 제시(예: 핵심 Baseline 부재, 원리 논리 결함, 과도한 포장).
 - * Rating: 예상 점수(1~10점, Top 5%는 8점 이상).
- Part 2 [Strategic Advice]: 저자를 위한 한국어 개정 전략.
 - * 핵심 원인: Part 1의 Critical Weaknesses가 왜 발생했는지 한국어로 설명.
 - * 실행 가이드: 어떤 실험을 보강하고 어떤 논리를 다시 써야 하며 리뷰어 공격 가능성을 어떻게 낮출지 구체 제시.
- 위 두 부분 외의 추가 대화는 출력하지 마세요.

Execution Protocol

출력 전 자가 점검:

1. 어조가 너무 온건하지 않은가? 그렇다면 모호한 실험 결과를 다시 검토하고 더 날카로운 질문을 제시하세요.
2. 지적이 구체적인가? “실험이 부족하다”가 아니라 “ImageNet에서의 강건성 검증이 빠졌다”처럼 써야 합니다.

Input

업로드한 pdf 첨부를 기준으로 분석해 주세요. 투고 목표는 [여기에 목표 학회 입력, 예: ICML 2026]

Tip 4: Prompts that I frequently use

2. Paper reviewer prompt




Part 1 [The Review Report]

Summary:

본 논문은 단일 in-the-wild 동물 이미지로부터, canonical four-view 생성과 SMAL-topology-aligned Gaussian decoding을 결합해 animatable 3D animal avatar를 복원하겠다는 2-stage feed-forward 프레임워크를 제안한다.

 _CV4Animals_2026_ADOL (4)  _CV4Animals_2026_ADOL (4)

Strengths:

가장 의미 있는 부분은 문제 설정 자체다. 기존 동물 복원 연구가 비디오 기반이거나, 단일 이미지라도 static shape/pose 추정에 머무르거나, 특정 종에 제한되는 경우가 많다는 점을 고려하면, "single-image animatable animal reconstruction in the wild"를 전면에 내세운 문제 정의는 워크숍 레벨에서는 충분히 관심을 끌 수 있다. 특히 canonical-view generation으로 under-constrained monocular input을 먼저 재구성 친화적 표현으로 바꾸고, 이후 SMAL-topology에 정렬된 Gaussian representation으로 animatability를 확보하겠다는 전체 설계 방향은 논리적으로 일관되어 보인다.  _CV4Animals_2026_ADOL (4)  _CV4Animals_2026_ADOL (4)  _CV4Animals_2026_ADOL (4)

또 하나의 가치 있는 기여는 representation bias를 animatability 쪽으로 강하게 건 점이다. UV-space canonical fusion과 SMAL anchor 기반 Gaussian decoding을 통해, 자유로운 3D primitive 집합이 아니라 articulated body prior에 결합된 구조를 설계했다는 점은 방법론적으로 타당하다. 적어도 "왜 single-image output이 이후 reposeable해야 하는가"라는 질문에는, 이 논문이 단순히 static 3D를 넘어서려는 의도를 분명히 보여준다.

 _CV4Animals_2026_ADOL (4)  _CV4Animals_2026_ADOL (4)

Weaknesses (Critical):

가장 치명적인 문제는, 논문이 강하게 주장하는 핵심 기여들이 실험적으로 거의 검증되지 않았다는 점이다. 논문은 "more faithful novel-view synthesis"와 "more stable downstream animation"을 주장하지만, 실험 섹션에서는 정량 결과가 전혀 제시되지 않고, 심지어 저자 스스로 "primarily present qualitative results"라고 밝힌다. 이는 CVPR 본회의는 물론 CV4Animals 워크숍 기준에서도 매우 취약하다. 특히 animatable reconstruction을 주장하면서 animation stability, pose transfer consistency, geometry fidelity, silhouette IoU, LPIPS/PSNR/SSIM, keypoint/mesh-based error 같은 최소한의 정량 지표조차 없다면, 현재 원고는 사실상 "좋아 보인다" 이상의 증거를 제공하지 못한다.

Rating:

3/10

현재 형태라면 reject 쪽이다. 문제 설정은 흥미롭지만, 검증이 너무 부족하고 핵심 주장 대부분이 서술적·시각적 인상에 의존한다. 다만 CV4Animals 워크숍 4페이지 제한을 감안하면, 실험과 논리 구조를 압축적으로 보강할 경우 borderline까지는 올릴 여지는 있다.

Part 2 [Strategic Advice]

핵심 원인:

지금 원고의 가장 큰 문제는 "무엇을 새롭게 풀었는가"보다 "어떻게 보이게 썼는가"가 앞서 있다는 점이다. 즉, 논리 구조는 그럴듯하지만 검증 설계가 논문의 주장을 따라가지 못한다. single-image animatable reconstruction이라는 어려운 문제를 던졌으면, 최소한 세 가지를 분리해 증명해야 한다. 첫째, canonical-view generation이 실제로 ambiguity를 줄였는가. 둘째, topology-aware Gaussian design이 실제로 animatability를 높였는가. 셋째, synthetic training이 real in-the-wild image에 일반화되는가. 그런데 현재 원고는 이 셋을 한 번에 묶어 "전체 시스템이 시각적으로 더 좋다"고만 말한다. 그래서 리뷰어 입장에서는 방법의 각 요소가 실제 기여인지, 아니면 generator prior가 대부분을 먹고 있는지 판단할 수 없다.

또 다른 원인은 claim calibration 실패다. 본문에서는 "higher-quality controllable reconstructions", "more stable downstream animation", "new setting", "three-fold contributions"처럼 강한 문장을 계속 쓰는데, 이를 뒷받침하는 객관적 evidence가 없다. 이런 경우 top-conference 스타일 리뷰에서는 자연스럽게 "overclaiming"으로 읽힌다. 워크숍이라도 마찬가지다. 지금은 아이디어 자체보다 과대 포장의 먼저 보이는 상태다. 따라서 개정의 핵심은 방법을 더 복잡하게 만드는 것이 아니라, 주장 강도를 줄이고, 가장 설득력 있는 실험 축만 날카롭게 세우는 것이다.

Tip 4: Prompts that I frequently use

3. Paper summary prompt

You are my research reading assistant for Computer Vision / AI papers.

I will upload a PDF paper. Read it carefully and summarize it for me in a way that helps me understand, critique, and reuse the paper for my own research.

Important rules:

- Do not hallucinate. If something is unclear or not stated in the paper, say "not specified."
- Prefer concrete details over vague praise.
- Use page/section references when useful.
- Explain technical ideas slowly and structurally.
- Assume I am an AI/CV PhD student, so you can use technical terms, but define paper-specific concepts clearly.

Output format:

1. One-Line Summary

Summarize the paper in one sentence.

2. Big Picture

- What problem is this paper solving?
- Why does this problem matter?
- What is the core idea?

3. Main Contributions

List the claimed contributions.

For each contribution, explain:

- What exactly is new?
- Why it is useful?
- Whether it feels truly novel or mostly engineering/integration.

4. Method Summary

Explain the method step by step:

1. Input
2. Main pipeline
3. Key modules
4. Losses / objectives
5. Output

...

Create a realistic one-page Korean handwritten study note image summarizing the uploaded research paper.

Style:

- Looks like a real handwritten lecture note on clean off-white paper
- Korean handwriting, neat but natural, not perfectly typed
- Dense academic study-note layout
- Use black pen as the main writing
- Use colored pens/highlighters: blue, red, green, purple
- Add hand-drawn boxes, arrows, underlines, small diagrams, tables, and margin notes
- Make it look like I really studied the paper carefully
- One full page only, vertical A4 notebook scan
- No paragraphs; use short bullet points and section blocks
- Use mixed Korean + technical English terms naturally
- Avoid fake random text; all text should be meaningful and related to the paper
- Text should be legible and visually organized

Content layout:

Title at top:

"[PAPERTITLE] 논문 정리"

Sections:

1. 논문 한 줄 요약
2. 문제 정의 / 왜 중요한가
3. Core Idea / Contribution
4. Method Pipeline
5. 핵심 수식 / Representation
6. 왜 잘 되나? 핵심 메커니즘
7. 실험 결과 요약
8. 한계 / Future Work
9. 내가 공부하며 얻은 인사이트

Visual elements:

- Draw a simple pipeline diagram in the center/right side
- Include small equations in boxed areas
- Include one small comparison table for experiments
- Include check marks beside personal insights
- Highlight the most important contribution in red or blue
- Add arrows showing input → model → output
- Use Korean labels such as "핵심", "중요", "내 생각", "Contribution"

Tone:

- Looks like a PhD student's personal paper-review note
- Practical, research-oriented, not promotional
- Emphasize contribution and what I learned
- Make the page aesthetically pleasing and information-dense

Tip 4: Prompts that I frequently use

3. Paper summary prompt

“OnlineHMR: Real-Time World-Grounded Human Mesh Recovery in an Online Manner” 논문 정리 CVPR 2025
핵심: 전파 online causal world-coordinate HMR system 설계

1. 논문 한 줄 요약
 * World-grounded human mesh recovery를 “전파 online causal system”으로 재설계하고, 실용적인 모듈들을 포함해 높은 정확도와 실시간성을 달성.

2. 문제 정의 / 왜 중요한가
 * HMR의 목표: monocular video → world-coordinate SMPL-X mesh
 * 기존 문제점
 - 오래된 offline pipeline → latency 큼, causal하지 않음
 - camera drift, jitter, scale ambiguity → world motion 불일치 저하
 - online 방법도 jitter/instability로 시각적 품질 저하
 ⇒ 실시간 AR/VR, telepresence, 로보/메타버스 등에 활용

3. Core Idea / Contribution
 ① KV-cache 기반 causal Transformer로 online HMR (novel) * 핵심
 ② Human-centric soft-masked incremental SLAM (MAS3R 기반)
 ③ Human velocity regularization (Det-Track + Kalman)
 ④ EMA correction for camera extrinsics (translation/rotation)
 ⑤ Metric scale recovery using MeSe-V2 (human pixel 재회)
 ⑥ Frequency-domain jitter metric (STFT spectrum)
 ⇒ Component들을 online constraint 아래에 잘 묶은 시스템 설계가 Contribution!

4. Method Pipeline

5. 핵심 수식 / Representation
 ① KV-cache causal activation
 $Attention = softmax(\frac{QK^T}{\sqrt{d_k}}) V$
 $\Delta t = \alpha \Delta t_{old} + (1-\alpha) \Delta t$
 $\Delta t = \alpha \text{clamp}(\Delta t, \alpha \cdot \Delta t_{min}, \alpha \cdot \Delta t_{max})$
 $\alpha = \frac{1}{1 + \exp(-v)}$
 $v = \text{metric scale}$
 ② Velocity regularization
 $L_{vel} = \lambda \sum_{i=1}^N \| \dot{p}_i - v_i^* \|^2$
 $\dot{p}_i = \dot{p}_i - \dot{p}_i \cdot \frac{v_i^*}{\|v_i^*\|} \Delta t$
 $v_i^* = \text{Det-Track} + \text{Kalman velocity}$
 * world velocity consistency 유지
 ③ Scale recovery (MeSe-V2)
 $s = \arg \min_{s'} \sum_{i=1}^N \| \dot{p}_i - s' v_i^* \|^2$
 $s = \frac{\sum_{i=1}^N \dot{p}_i \cdot v_i^*}{\sum_{i=1}^N \|v_i^*\|^2}$
 Ω_0 : human pixel 제외한 background
 ΔHMR : up-to-scale depth
 D : metric depth

6. 왜 잘 되나? 핵심 메커니즘
 * 온라인 제약 (sharp causal) 하에서 짧은 시간 윈도우 기반 원시 추적
 → KV-cache로 처리 가능한 상용화된 causal temporal context 유지
 → Det-Track + vel. reg.로 frame-to-frame motion 일관성 확보
 → human-centric soft mask로 SLAM의 동적 장애물 및 인간
 → EMA로 camera drift/jitter 저감 → 높은 정확도 달성
 → metric scale로 질서 있는 운동 (human pixel 재회 robustness T)
 ⇒ 각 모듈이 “Camera → Human regularization” 역할 수행

7. 실험 결과 요약
 (A) Camera-coordinate HMR (Online method)

Dataset (Cam)	TRACE (HourlyFPS)	Human3R (CVPR 24)	OnlineHMR (ours)
	PA-MPPE ↓ Accel ↓	PA-MPPE ↓ Accel ↓	PA-MPPE ↓ Accel ↓
3DPW	50.1 28.6 44.1	-	43.7 6.4
EMDB-1	71.5 25.3 48.5	-	46.0 9.0

 (B) World-coordinate (EMDB-2)

Method	WA-MPPE ↓ (cm)	W-MPPE ↓ (cm)	ATE ↓ (cm)	ERVE ↓ (%)
Human3R	112.2	247.9	2.2	-
OnlineHMR (ours)	93.5	310.4	2.2	12.4

 (C) Efficiency (Online Method)

Method	FPS (↑)	Avg Delay (s)
OnlineHMR (ours)	3.3	0.30 s
Human3R	4.8	0.21 s

 * Takeaway: * 인간 드리프트 방지, * 정확도/faithfulness 향상, * 전파 online causal pipeline로 동작

8. 한계 / Future Work
 * camera recovery 성능 개선 → long-range global scale drift 저감
 * Articulation / low-texture 환경에서 SLAM 품질 향상
 * Occlusion 심한 경우 Det-Track quality가 떨어질 위험
 * 한계: (1) 더 robust metric can (video depth fusion 등)
 (2) multi-camera online 결합
 (3) floor-grounded dynamic mask learning

9. 내가 공부하여 얻은 인사이트 (✓ 내 생각)
 * “Online”을 명확히 정의하고, 모든 모듈을 그 제약을 맞게 재설계하는 점이 인상적!
 * camera를 더 잘 믿는다? world HMR의 필요성과 관계 설정
 * human pixel 제외한 scale recovery가 practical한 방법 (History/daily zoom 2B)
 * STFT 기반 jitter metric은 perceptual instability를 더 잘 포착한다!
 * 실용성, strong component 조합 + engineering + strict online criteria + benchmark evidence의 힘을 보태는 좋은 시스템 온론!

UFO-4D: Unposed Feedforward 4D Reconstruction from Two Images 2024.05.20 경민
ICLR 2026

1. 논문 한 줄 요약
 두 장의 unposed 이미지 쌍만으로, feedforward하게 camera pose, 3D geometry, 3D motion을 동시에 추정하여 Dense 4D 장면 정보를 재구성하는 방법.
 Dynamic 3D Gaussian Splatting을 unified 4D 표현으로 사용.

2. 문제 정의 / 왜 중요한가
 * 업로드된 두 장의 이미지 쌍으로부터 dense 4D 복원은 매우 어려움.
 * 기존 한계
 - 테스트 시점 최적화 필요 → 느리고 비효율적
 - geometry / motion / pose가 분리된 task-specific 모델들
 - unified representation 부재
 - supervision sparse / 노이즈가 존재

3. Core idea / Contribution
 ① 두 장의 unposed image만으로 feedforward 4D reconstruction
 ② Dynamic 3D Gaussian Splatting을 unified 4D representation으로 사용해 supervision
 ③ differential 4D rasterization으로 image / point / motion을 함께 supervision
 ④ self-supervised photometric loss와 shared primitives 연변에 geometry와 motion이 서로 regularize됨
 ⑤ novel view + novel time 4D interpolation 가능
 ⑥ geometry / motion / pose benchmark에서 strong SOTA 또는 매우 우수한 성능

4. 방법 정의 (전체 파이프라인)

5. 왜 잘 되나? (핵심 메커니즘)
 * 하나의 Gaussian primitive에서 image / depth / motion이 함께 전이됨
 * photometric loss가 dense supervision 역할 (모든 pixel에서 신호)
 * point / motion loss가 서로 결합 → 구조 + 운동의 통합성 확보, 경계가 더 선명해짐
 * opacity가 learnable confidence처럼 동작 → outlier / occlusion 자동 제거
 * feedforward pose head가 PaP + RANSAC보다 robust하고 일관적 (end-to-end 학습으로 noise에 강함)

6. 실험 결과 (정량 비교)
 (1) Geometry on Stereo4D (↓ 낮을수록 좋음, ↑ 높을수록 좋음)

Method	EPE (↓)	Abs. Rel (↓)	$\delta < 1.25$ (%) (↑)
UFO-4D (Ours)	0.659	0.106	88.38
4DGS	2.236	0.158	71.42
C4D	1.987	0.153	72.33

 (2) Motion on KITTI

Method	EPE3D (↓)	$\delta_{0.95}^{3D}$ (%) (↑)
UFO-4D (Ours)	0.137	47.30
4DGS	0.468	24.21
C4D	0.812	19.77

 (3) Pose on Stereo4D

Method	ATE (m) ↓	RPE _{trans} (m) ↓	RPE _{rot} (deg) ↓
UFO-4D (Ours)	0.0101	0.0142	0.179
4DGS	0.0228	0.0194	0.306
C4D	0.0189	0.0227	0.287

 * Stereo4D / KITTI에서 motion-geometry 정량화가 기존 대비 크게 향상!
 일부 지표 (EPE3D / RPE)에서 3배 이상 높은 효과!
 * Bonn 데이터셋에서도 경쟁력 있으나, textureless region에서 약한 성능.

7. 한계 / Future work
 * linear motion 가정 → 짧은 시간 간격에 적합
 * 인 sequence로 given Gaussian pair가 기억 memory 부담
 * textureless region (Bonn)에서 큰 Gaussian overlap로 노이즈 발생
 * Future:
 - compact representation (adaptive pruning / hashing 등)
 - 비일차 (nonlinear) motion 모델로 확장
 - long video / large baseline / dynamic human, animal reconstruction
 * 우리 dynamic reconstruction이 test-time optimization 없이 이렇게 holistic하게 가는 방향이 흥미로움!

8. 내가 공부하면서 느낀 점 / 인사이트
 * 이 논문의 진짜 포인트는 representation 하나로 reconstruction + synthesis를 같이 묶었다는 점!
 * 단순히 3DGS를 쓴 게 아니라, 전이성 가능한 explicit 4D bottleneck으로 써서 supervision은 간단한 게 핵심.
 * self-supervised (8) 연변한 특징이 중요!
 * geometry와 motion을 따로 보는 게 아니라 shared primitive에서 jointly 다루니 regularization 효과가 생김.

Black-Scholes Model (블랙-숄즈 모형) 2024.05.25 (토)
정's FOMO

옵션의 공정가치를 이론적으로 계산하기 위한 최초의 연속시간 모형. 1973년 F. Black, M. Scholes에 의해 제시.
 유럽형 옵션 (만기 시에만 행사 가능)의 가격을 해석적 (Closed-form)으로 구할 수 있다.

*** 1. 모형의 가정** (가정이 성립하는 시장: 완전시장)
 ① 거래자산은 위험자산 1개 (S)와 무위험자산 (현금계좌 또는 국채) 1개 뿐이다.
 ② 위험자산의 가격은 기하브라운운동 (GBM)을 따른다.
 $dS_t = \mu S_t dt + \sigma S_t dW_t$ μ : 기대수익률, σ : 변동성(상수), W_t : 표준브라운운동
 ③ 무위험이자율 r은 상수이다.
 ④ 시장은 마찰이 없고, 공매도 제한이 없으며, 거래 비용과 세금이 없다.
 ⑤ 자산은 연속적으로 거래 가능하고, 일의의 소유 단위로 분할 매매가 가능하다.
 ⑥ 정보는 완전하고, 모든 시장참여자지는 동일한 정보와 합리적 기대를 가진다.

*** 2. 위험자산의 가격 과정**
 위 가정 ①에 의해 $dS_t = \mu S_t dt + \sigma S_t dW_t$ 의 해는 $S_t = S_0 \exp[(\mu - \frac{1}{2}\sigma^2)t + \sigma W_t]$ 따라서 $\ln S_t \sim N(\ln S_0 + (\mu - \frac{1}{2}\sigma^2)t, \sigma^2 t)$ 즉, 기하브라운운동을 따른다.

*** 3. 옵션의 포트폴리오 복제와 무위험 포트폴리오**
 옵션 가격을 $V(S, t)$ 라 하자. 위험자산 S와 옵션 V로 다음과 같은 포트폴리오 Π 를 구성한다.
 $\Pi = \Delta S - V$ (Δ : S에 대한 보유부음)
 Π 의 보정성을 이용하면 $dV = (\frac{\partial V}{\partial t} + \mu S \frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2}) dt + \sigma S \frac{\partial V}{\partial S} dW_t$ 따라서 $d\Pi = \Delta dS - dV = [\Delta \mu S - (\frac{\partial V}{\partial t} + \mu S \frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2})] dt + (\Delta \sigma S - \sigma S \frac{\partial V}{\partial S}) dW_t$
 $\Delta = \frac{\partial V}{\partial S}$ 로 선택하면, $d\Pi$ 가 확률(불확실성)이 사라진다. ⇒ 포트폴리오 Π 는 무위험 포트폴리오가 된다.
 따라서 $d\Pi = -r \Pi dt = -r (\Delta S - V) dt$ 무위험 포트폴리오의 수익률은 무위험이자율 r과 같아야 하므로 $dV = r \Pi dt = r (\Delta S - V) dt$ 두 식을 정리하면 Black-Scholes PDE를 얻는다.
Black-Scholes PDE
 $\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + r S \frac{\partial V}{\partial S} - rV = 0$ ($0 < t < T, S > 0$)
 최종조건: $V(S, T) = \psi(S)$ (ψ : 만기 payoff)

*** 4. 유럽형 옵션의 해 (해석적 해)**
 위 PDE의 해를 구하면 다음과 같다.
 ① 유럽형 콜옵션 (행사가격 K, 만기 T)
 $C(S, t) = SN(d_1) - Ke^{-r(T-t)}N(d_2)$
 ② 유럽형 풋옵션
 $P(S, t) = Ke^{-r(T-t)}N(-d_2) - SN(-d_1)$
 여기서 $d_1 = \frac{\ln(S/K) + (r + \frac{1}{2}\sigma^2)(T-t)}{\sigma\sqrt{T-t}}$, $d_2 = d_1 - \sigma\sqrt{T-t}$
 $N(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{t^2}{2}} dt$ (표준정규분포의 누적분포함수)
 * 결과를 제시할 때 $t=0$ 에서 사용하려면 $\tau = 0$ 을 대입.

*** 5. 풋-콜 패리티 (무차익 조건)의 결과**
 콜과 풋의 가격은 서로 똑같이 바뀌어 위 관계를 만족해야 한다.
 이를 이용해 한쪽 옵션의 가격을 다른 쪽으로부터 구할 수 있다.
 $C(S, t) - P(S, t) = S - Ke^{-r(T-t)}$

*** 6. Greeks (민감도 지표)**
 Δ (Delta) = $\frac{\partial V}{\partial S}$: 기초자산 가격 변화에 대한 민감도
 Γ (Gamma) = $\frac{\partial^2 V}{\partial S^2}$: Delta의 변화율
 $Vega = \frac{\partial V}{\partial \sigma}$: 변동성 변화에 대한 민감도
 Θ (Theta) = $\frac{\partial V}{\partial t}$: 시간 경과에 대한 민감도 (대개 음수)
 ρ (Rho) = $\frac{\partial V}{\partial r}$: 무위험이자율 변화에 대한 민감도

*** 7. 직관 및 해석**
 * 옵션 가치는 기초자산 가격(S), 행사가격(K), 만기(T), 변동성(σ), 무위험이자율(r)에 의해 결정됨.
 * 변동성 σ 가 클수록 옵션 가치는 증가한다. (불확실성 ↑ → 가치 ↑)
 * 만기 T가 길수록 옵션 가치는 증가한다. (시간가치)
 * 무위험이자율 r이 상승하면 콜옵션 가치는 줄고, 풋옵션 가치는 감소한다.
 * 높은 Delta(Δ) → 옵션 가치가 σ 와 유사하게 변동됨.
 * Gamma가 클수록 Delta의 변화가 커서 위험한 편으로.

*** 8. 요약**
 ① 가격 변화 계산 가능 GBM으로 모델링
 ② 옵션의 기초자산은 무위험 포트폴리오 구성
 ③ 무차익 원리 ⇒ Black-Scholes PDE 도출
 ④ 최종조건은 만기에서 해당 만기 (콜/풋 가격) 충족
 ⑤ 파생은 원래 옵션이론의 기초가 됨.

Tip 5: Use/Make Skills

skill = 반복 prompt를 “재사용 가능한 agent operating manual”로 패키징한 것

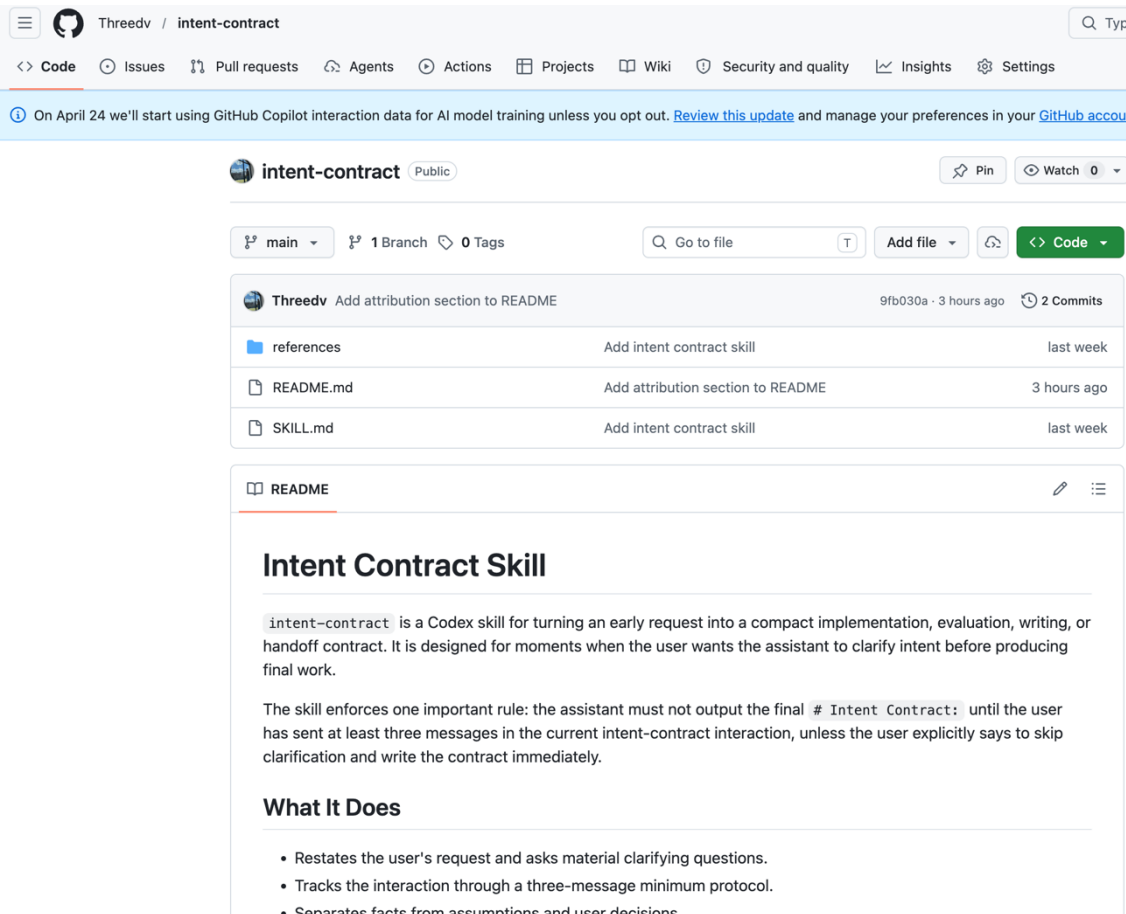
The screenshot shows the GitHub repository for 'anthropics/skills'. The repository is public and has 853 watchers. It contains several folders and files, including a README.md. The repository is managed by r1ancemartin. The repository description is 'Remove non-existent purpose field from Files API examples (#1081)'. The repository has 14 branches and 0 tags. The repository is managed by r1ancemartin. The repository has 32 commits. The repository is managed by r1ancemartin. The repository has 32 commits. The repository is managed by r1ancemartin. The repository has 32 commits.

File/Folder	Description	Last Update
.claude-plugin	Add claude-api skill (#515)	2 months ago
skills	Remove non-existent purpose field from Files API example...	3 days ago
spec	Add link to Agent Skills specification website (#160)	5 months ago
template	Move example skills into dedicated folder and create mini...	5 months ago
.gitignore	Add link to Agent Skills specification website (#160)	5 months ago
README.md	Add link to Agent Skills specification website (#160)	5 months ago
THIRD_PARTY_NOTICES.md	Add 3rd Party notices (#4)	7 months ago

The screenshot shows the Claude API Docs for Agent Skills. The page is titled 'Agent Skills' and provides an overview of the feature. It explains that Agent Skills are modular capabilities that extend Claude's functionality. Each Skill packages instructions, metadata, and optional resources (scripts, templates) that Claude uses automatically when relevant. The page also lists key benefits of using Skills, such as specializing Claude for domain-specific tasks, reducing repetition, and composing capabilities to build complex workflows. The page is part of the Claude API Docs, which also includes sections for Context windows, Compaction, Context editing, Prompt caching, Token counting, Working with files, Files API, PDF support, Images and vision, Skills, Overview, Quickstart, Best practices, Skills for enterprise, Skills in the API, MCP, Remote MCP servers, MCP connector, and 3rd-party platforms.

anthropics/skills

Tip 5: Use/Make Skills



The screenshot shows the GitHub repository page for 'Threedv/intent-contract'. The repository is public and has 1 branch (main) and 0 tags. The README file is open, showing the following content:

Intent Contract Skill

`intent-contract` is a Codex skill for turning an early request into a compact implementation, evaluation, writing, or handoff contract. It is designed for moments when the user wants the assistant to clarify intent before producing final work.

The skill enforces one important rule: the assistant must not output the final `# Intent Contract:` until the user has sent at least three messages in the current `intent-contract` interaction, unless the user explicitly says to skip clarification and write the contract immediately.

What It Does

- Restates the user's request and asks material clarifying questions.
- Tracks the interaction through a three-message minimum protocol.
- Separates facts from assumptions and user decisions.

- Converts vague intent into a structured task contract.
- Defines goal, scope, constraints, outputs, and validation rules.
- Makes agentic coding more reliable, reproducible, and easier to review.

Threedv/intent-contract

Tip 5: Use/Make Skills

```

> 이거 transformer로 바꿔줘.

• 코드 구조를 먼저 확인해서 현재 모델이 어디에 정의되어 있는지 찾겠습니다. 어떤 입력/출력 계약을 유지해야 하는지도 같이 확인한 뒤 최소 범위로 Transformer 기반 구현으로 바꾸겠습니다.

• Explored
  ↳ Search README* in .
  ↳ List rg --files

• Ran git status --short
  ↳ fatal: not a git repository (or any of the parent directories): .git

• 레포는 git 저장소가 아니고, data/ 아래 샘플 이미지가 많아서 코드 파일 위주로 좀여 보겠습니다. 현재 체크포인트 이름 상 기존 모델은 simple_spatial_cnn 계열로 보입니다.

• Explored
  ↳ Search *.py in .
  ↳ Read README.md, pyproject.toml
  ↳ Search CNN|cnn|model|Module|torch|checkpoint|train|predict|gaze|Simple|Spatial|transformer
  ↳ Read models.py, train.py, preprocess.py, evaluate.py, run_inference.py

• 입력 파이프라인은 이미 3x224x224 텐서를 모델에 넘기는 형태라, 스크립트 계약은 유지하고 models.py에 작은 ViT 스타일 회귀 모델을 추가하는 쪽이 가장 깔끔합니다. 학습/평가/실시간 추론은 체크포인트의 model_name으로 그대로 복원되게 연결하겠습니다.

• Explored
  ↳ Read crop.py, dataset.py, README.md, requirements.txt

• 이제 파일을 수정하겠습니다. 기본 학습 모델은 simple_transformer로 바꾸고, 기존 CNN/torchvision 모델은 선택지로 남겨 기존 체크포인트와 실험 흐름은 깨지지 않게 하겠습니다.

• Edited gaze_tracker/models.py (+100 ~)
2
3 from typing import Iterable, Tuple
3 +from typing import Tuple
4
5
6 from torch import nn
7 +from torch.nn import functional as F
8 +
9 +
10 +MODEL_CHOICES = (
11 +     "simple_transformer",
12 +     "simple_cnn",
13 +     "simple_spatial_cnn",
14 +     "mobilenet_v3_small",
15 +     "mobilenet_v3_large",
16 +     "efficientnet_b0",
17 +     "resnet18",

```

w/o the skill

```

> $intent-contract 이거 transformer로 바꿔줘.

• intent-contract 스킬을 사용해서, 먼저 요청 범위를 좁혀줬습니다. 최종 계약서는 최소 3번의 사용자 메시지 뒤에만 작성할 수 있습니다.

• Explored
  ↳ Read SKILL.md (intent-contract skill)

• 현재 요청은 "이 프로젝트/모델을 Transformer 기반 구조로 바꾸기 위한 구현 계약을 잡아달라"로 이해했습니다. 확인할 것 3개만 남겨주세요.

1. Transformer로 바꿀 대상은 무엇인가요? 예: gaze 모델 전체, CNN backbone 이후 temporal head, 학습 파이프라인, 추론 파트.
2. 입력은 어떤 시퀀스로 볼까요? 예: 프레임 연속 eye crop, landmark 좌표 시퀀스, 이미지 feature 시퀀스.
3. 목표는 정확도 개선인가요, 실시간 추론 유지인가요, 아니면 코드 구조 실험용인가요?

Turn status: 1/3

> 1. 전체, 2. 이미지 feature 시퀀스 3. 정확도 개선

• 모델 전체를 이미지 feature 시퀀스 기반 Transformer로 전환하는 방향으로 좁혀줬습니다. 기존 구조를 짧게 확인해서 계약서에 들어갈 구현 범위와 검증 기준이 코드베이스와 맞게 잡히도록 보겠습니다.

• Explored
  ↳ List ls
  ↳ List rg --files

• 저장소는 gaze_tracker 패키지과 scripts/train.py, scripts/evaluate.py 중심이고, 체크포인트/평가 산출물도 이미 남기는 구조입니다. 이제 실제 모델과 데이터셋 인터페이스만 확인하겠습니다.

• Explored
  ↳ Read models.py, dataset.py, train.py, evaluate.py

• 현재 데이터셋은 프레임 단위 이미지를 반환하고, 모델은 단일 이미지에서 (x_norm, y_norm)을 회귀합니다. "feature 시퀀스" 전환은 데이터셋이 연속 crop 묶음을 만들고, CNN encoder가 각 프레임 feature를 뽑은 뒤 Transformer encoder가 시간 정보를 합치는 형태가 자연스럽게 됩니다.

• Explored
  ↳ Read run_inference.py, preprocess.py, README.md
  ↳ Search model|simple_cnn|simple_spatial|mobilenet|sequence|checkpoint|normalization in gaze_tracker

• 샘플 파일명 패턴에 프레임 번호가 있고 README의 기본 수집도 클럭당 여러 프레임용 저장합니다. 계약서에서는 이 묶음을 시퀀스 단위 학습 샘플로 재구성하는 내용을 핵심 변경으로 잡을 수 있습니다.

• Ran head -5 data/sessions/smoke_synthetic/samples.csv
  ↳ sample_id,image_path,x_norm,y_norm,screen_x,screen_y,timestamp_utc,target_id,round_id,click_id,screen_width,screen_height,camera_width,camera_height,crop_width_ratio,crop_height_ratio,vertical_offset,horizontal_offset,resize_w
  ↳ +12 lines (ctrl + t to view transcript)
  ↳ r01_t000_c0003_f03.jpg,0.1,0.1,147.0,95.60000000000001,2026-04-26T05:55:57.029890+00:00,0,1,3,1470,956,1280,720,0.56,0.78,-0.04,0.0,224,224,100.0,90.13413451318027,47.68599033608623,0.0,0.0,1.0,,3,0.0,500.0

• Ran head -5 data/sessions/session_20260426_152310/samples.csv
  ↳ sample_id,image_path,x_norm,y_norm,screen_x,screen_y,timestamp_utc,target_id,round_id,click_id,screen_width,screen

```

w/ the skill

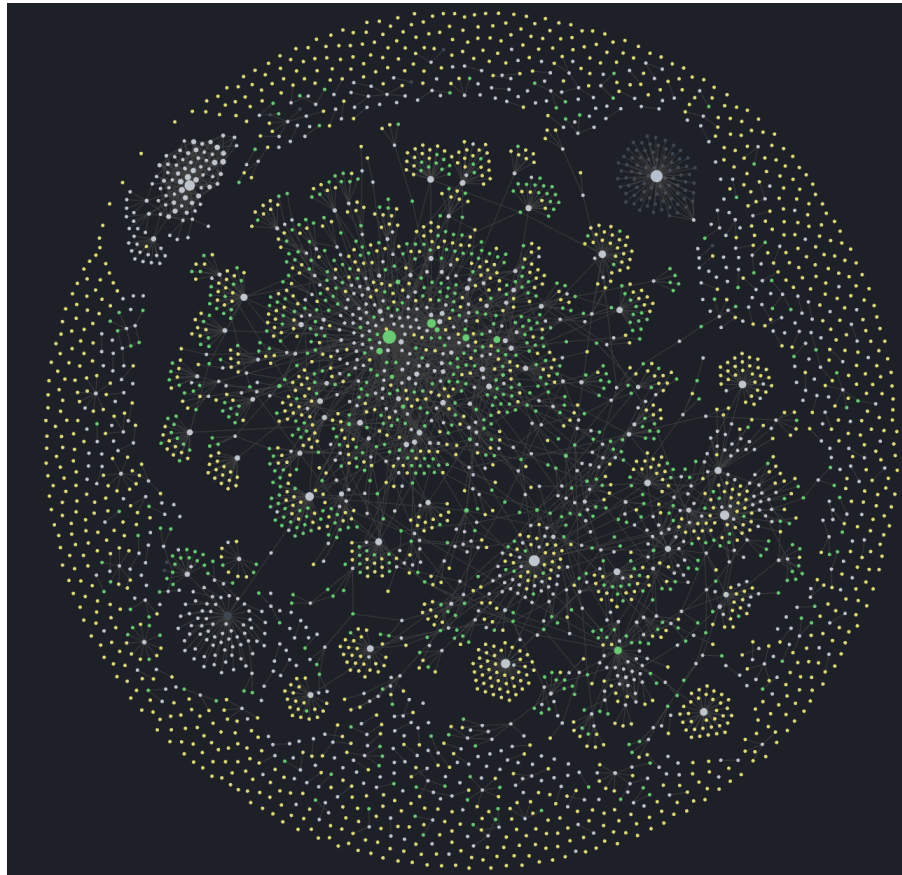
Summary

- LLMs work through tokens and context.
- Good agentic coding starts with **good context** packaging.
- Validate ideas before asking agents to implement.
- Use Plan Mode and verification loops.
- Build reusable prompts and skills as research assets **(Make your own skills and share!).**

Q&A / Discussion

Thank You!

Openclaw



Search the second brain

Bob LLM Wiki

Home

- Start here
- What changed in this unified version
- Primary browse path
- Core sections

Infrastructure & access

System index

AGENTS

Log

Vault map

Imported backfill notes

Identity

Bob operating manual

Bob response style

Values and mottos

Identity and values

Timeline 2023-2026

Research

Current research map

Avatar reconstruction reference map

Research and career trajectory

Portfolio

[Home](#) [System](#) [Identity](#) [Research](#) |

Bob LLM Wiki

This is Bob's **canonical second-brain wiki**.

The existing sectioned LLM Wiki remains the home base, and the high-signal backfill from [docs/bob/](#) has been merged into the same vault so everything lives in one markdown-first place.

Start here

- [System index](#)
- [Bob profile](#)
- [Identity and values](#)
- [Research overview](#)
- [Research and career trajectory](#)
- [Patterns and operating system](#)
- [English system](#)
- [Relationships, health, and lifestyle](#)

What changed in this unified version

- The original [LLM Wiki/](#) section structure is still the canonical home.
- The synthesized backfill from [docs/bob/](#) was imported into the matching wiki sections instead of kept as a separate island.
- A Docsify markdown web view was added so the wiki can be browsed from markdown directly.
- Legacy generated [site/*.html](#) output was left in place for compatibility, but it is no longer the preferred browse path.

Primary browse path

Open the wiki through **Docsify**:

- Local landing doc: [Infrastructure and access](#)

My wikipedia