

000  
001  
002  
003 

# 4D LATENT WORLD MODEL FOR ROBOT PLANNING

004  
005  
006  
007  
008  
009  
010  
011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
054  
055  
056  
057  
058  
059  
060  
061  
062  
063  
064  
065  
066  
067  
068  
069  
070  
071  
072  
073  
074  
075  
076  
077  
078  
079  
080  
081  
082  
083  
084  
085  
086  
087  
088  
089  
090  
091  
092  
093  
094  
095  
096  
097  
098  
099  
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  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1098  
1099  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1198  
1199  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1298  
1299  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1398  
1399  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1498  
1499  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1598  
1599  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1698  
1699  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1798  
1799  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1898  
1899  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1909  
1

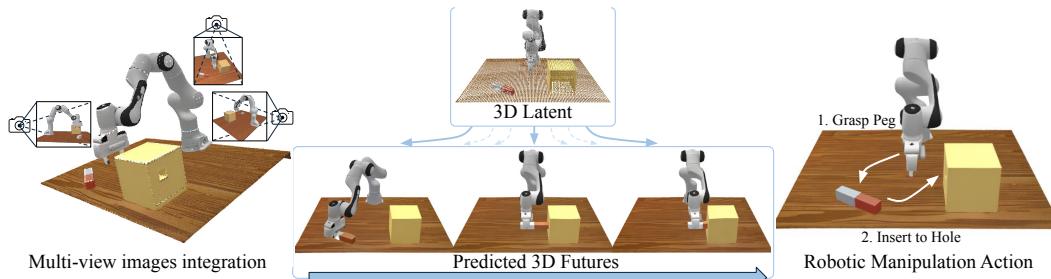


Figure 1: Our 4D latent world model integrates multi-view images and text instructions to forecast future 3D dynamics, enabling robots to plan and execute tasks that require precise 3D understanding.

tasks demand more than recognition of visual appearance: they require an accurate understanding of 3D geometry, object pose, and spatial relationships. This fundamental challenge exposes a key limitation of modern robot learning.

Modeling the dynamics of a scene directly in 3D, however, is a challenging task. Traditional 3D representations, such as point clouds and meshes, preserve geometry but lose rich visual detail necessary for semantic understanding. Photorealistic representations like Neural Radiance Fields (NeRFs) (Mildenhall et al., 2021) or 3D Gaussian Splatting (Kerbl et al., 2023) better capture appearance, but are computationally intensive and not easily amenable to dynamic modeling. A common compromise is to predict RGB video along with depth and normals Zhen et al. (2025), which provide partial 3D cues but still reduce to surface-level projections, leaving them vulnerable to occlusions and viewpoint shifts. This leads to a question, *Can we build a model that inherently simulates dynamic 3D structures of the world?*

In this paper, we propose a **4D latent world model** for robot planning. Following the success of Latent Diffusion Models (Rombach et al., 2022) which utilize spatially-aware 2D feature maps rather than unstructured 1D global latents, we adopt a structured 3D latent representation (Xiang et al., 2025) for 3D scenes. Specifically, we encode the scene into a sequence of sparse voxel grids where active voxel holds a compact feature vector. The *grid latent* design allows us to maintain explicit 3D spatial biases, while benefiting from the computational efficiency and semantic abstraction of a low-dimensional latent space. Based on the structured 3D latents, our model learns the dynamics for the 3D scene and generates plausible future latents conditioned on current observations and text instructions. Unlike methods restricted to surface maps or videos, our latent captures holistic 3D information of the scene that can be decoded into various formats, such as point clouds or 3D Gaussian Splatting. This approach enables our world model to achieve a more complete understanding of 3D structures and generate futures with superior physical and spatial consistency. This detailed 3D information is then leveraged by a goal-conditioned inverse dynamics model, which translates the generated futures into precise robot actions and is especially effective for fine-grained, 3D-aware tasks. In summary, our contributions are as follows:

- i) We introduce a 4D latent world model that predicts future 3D structures conditioned on current observations and text goal, achieving high visual quality, physical consistency, and robust viewpoint generalization.
- ii) We propose a planning framework that leverages our model’s detailed 3D predictions as geometrically rich goals for an inverse dynamics controller, enabling precise and spatially-aware manipulation.
- iii) Experiments demonstrate that our method outperforms state-of-the-art robot world models in both generation quality and downstream robotic task performance, including strong zero-shot generalization to various visual changes, and effective transfer to a real-world robot task.

## 2 RELATED WORK

**General Purpose Embodied Models.** A dominant paradigm in robotic learning and embodied agents has been the development of large multitask policies that directly map sensory inputs to output actions. Through the collection of large-scale multi-task embodied and robotics datasets, such models (Reed et al., 2022; Lee et al., 2022; Huang et al., 2023; Zitkovich et al., 2023; Kim et al., 2024; Barreiros et al., 2025; Hou et al., 2025; NVIDIA et al., 2025; Black et al., 2024) are able to

108 solve tasks across many environments. However, there are two large challenges with constructing  
 109 such general-purpose policies across many environments. First, the action space across environments  
 110 is often misaligned, with existing works requiring careful action tokenization (Reed et al.,  
 111 2022), and second, small changes in the environment cause policies to fail. To circumvent these is-  
 112 sues, our work focuses on learning a 3D model of the world and then planning on top of the model to  
 113 act in the environment. This approach allows us to use a shared underlying 3D state of the world to  
 114 transfer across environments. At the same time, by learning the more complex task of modeling the  
 115 3D dynamics of the world, we are able to effectively generalize across many environmental changes.

116 **Generative World Models for Planning.** Learned world models have recently been explored for  
 117 robot planning, often through video prediction from a single viewpoint (Janner et al., 2022; Ajay  
 118 et al., 2022; Li, 2023; Ajay et al., 2023; Du et al., 2023a; Ko et al., 2023; Yang et al., 2023; Li  
 119 et al., 2023; He et al., 2023; Alonso et al., 2024; Chen et al., 2024; Ubukata et al., 2024; Bar et al.,  
 120 2025; Qi et al., 2025; Xie et al., 2025a). For example, UniPi (Du et al., 2023b) frames planning  
 121 as generating a video trajectory, which improves interpretability but lacks explicit 3D structure,  
 122 leading to inconsistencies under occlusion or viewpoint change. To address this, hybrid approaches  
 123 such as TesserAct (Zhen et al., 2025) extend video models to predict future depth and normal maps,  
 124 providing stronger spatial priors for manipulation. **However, these methods are fundamentally 2.5D**  
 125 **and operate in pixel space, which remain surface-level projections that struggle to maintain full**  
 126 **multi-view coherence.** In contrast, our method models dynamics directly in a 3D latent space, **which**  
 127 **enables inherent 3D modeling rather than relying solely on 2.5D projections**, ensuring consistent  
 128 multi-view rollouts and providing geometrically grounded subgoals.

129 **3D Dynamics and Planning with Explicit Geometry.** A parallel line of research learns dynamics  
 130 over structured 3D representations such as point clouds, meshes, or NeRF-like fields, enabling  
 131 physical simulation or relational reasoning for manipulation tasks. These include behavior-primitive  
 132 dynamics for stowing (Chen et al., 2023), point-cloud relational planning (Huang et al., 2025), and  
 133 deformable-object digital twins (Jiang et al., 2025). Similarly, graph-based dynamics have been  
 134 applied to elasto-plastic manipulation (Shi et al., 2023; 2022) and latent relational planners (Huang  
 135 et al., 2024), while others utilize compositional NeRFs for multi-object scenes (Driess et al., 2023).  
 136 While these methods succeed in specific domains, they typically rely on object-centric factoriza-  
 137 tions, pre-defined primitives, or task-specialized graph structures. In contrast, our approach learns  
 138 a holistic latent 3D scene representation. It aggregates multi-view geometry into a unified state,  
 139 supports 4D rollouts directly in latent space, and decodes into diverse 3D formats (point clouds  
 140 or multi-view images rendered from 3D Gaussians). This formulation allows our model to jointly  
 141 model dynamics and planning in a unified framework while maintaining flexibility across tasks,  
 142 without requiring predefined object primitives or action parameterizations.

### 3 FORMULATION OF LATENT WORLD MODELING

#### 3.1 PROBLEM FORMULATION

146 Our goal is to build a 4D world model that learns the dynamics of a 3D environment over time. We  
 147 formalize it as a conditional generator  $g(\mathbf{o}_{t+1}, \dots, \mathbf{o}_{t+T} | \mathbf{o}_t, a)$ . Here, given the state of the 3D scene  
 148  $\mathbf{o}_t$  at time  $t$  and an action  $a$ , the model predicts a sequence of future 3D scene states  $\{\mathbf{o}_{t+1}, \dots, \mathbf{o}_{t+T}\}$   
 149 over a horizon  $T$ .

150 In practice, the complete 3D scene  $\mathbf{o}_t$  is not directly observable. Instead, it is only seen through par-  
 151 tial observations  $\{\mathbf{o}_t^{(i)}\}$ , such as RGB or depth images from multiple cameras in real-world setups,  
 152 or renderings from simulated viewpoints. These observations must be geometrically consistent, as  
 153 they all describe the same underlying 3D structure  $\mathbf{o}_t$ . The action  $a$  can range from a low-level  
 154 control signal to a high-level semantic instruction. In this work, we focus on text-based instruc-  
 155 tions that specify the desired evolution of the agent and the environment. **In our implementa-  
 156 tion, language instructions serve as the high-level action input that guides the latent rollout, while the  
 157 inverse dynamics module produces the low-level robot commands as absolute joint positions.**

158 Prevailing world modeling methods are primarily based on video generation, predicting se-  
 159 quences of 2D frames (sometimes augmented with depth and normals) from a single viewpoint  
 160  $g^{(i)}(\mathbf{o}_{t+1}^{(i)}, \dots, \mathbf{o}_{t+T}^{(i)} | \mathbf{o}_t^{(i)}, a)$ . A straightforward extension to 3D is to train separate world models for  
 161 each viewpoint and then fuse their outputs. However, such designs do not naturally support true

162 4D world modeling. Instead, we introduce a *4D latent world model* that directly addresses the key  
 163 requirements:

- 165 • **3D consistency:** By encoding the complete 3D scene at timestep  $t$  into a single holistic latent  
 166 representation  $z_t$ , our model ensures that predictions across views adhere to the same underlying-  
 167 ing 3D structure.
- 168 • **Multi-view reasoning:** The shared latent aggregates information from multiple observations,  
 169 allowing cues from one view to inform predictions in others.
- 170 • **Flexible generalization:** The latent can be decoded into diverse explicit 3D formats (e.g., point  
 171 clouds, 3D Gaussians), allowing the framework to adapt to novel viewpoints and various scene  
 172 representations.

173 Together, these properties enable a unified 4D latent world model that predicts future latent states,  
 174  $g(z_{t+1}, \dots, z_{t+T} | z_t, a)$ . The latent  $z_t$  is designed to be decodable into various explicit 3D represen-  
 175 tations, such as point clouds or 3D Gaussians, which allows use to obtain any desired observation  
 176  $o_t^{(t)}$  decoded from the state.

### 178 3.2 3D LATENT FOR SCENE REPRESENTATIONS

180 Our world model requires a 3D latent representation  $z$  that is both compact enough for efficient  
 181 dynamic modeling and expressive enough to capture the fine details of the complete 3D structure.  
 182 Traditional representations, such as meshes, point clouds, or SDFs, often lack photorealism, while  
 183 modern representations, like NeRFs (Mildenhall et al., 2021) or 3D Gaussians Kerbl et al. (2023),  
 184 are computationally expensive to generate directly at every timestep.

185 To balance efficiency and expressivity, we adopt a structured, sparse latent representation inspired  
 186 by SLAT (Xiang et al., 2025). Our latent scene representation  $z_t$  is defined as a set of sparse voxel  
 187 features:  $z_t = \{(p_i, f_i)\}_{i=1}^L$ . Here, within a discretized  $N \times N \times N$  grid of the 3D scene,  $p_i \in$   
 188  $\{0, 1, \dots, N-1\}^3$  denotes the 3D coordinate of one of the  $L$  active voxels, and  $f_i \in \mathbb{R}^d$  is a feature  
 189 vector encoding local geometry and color. **This representation balances structural information with**  
 190 **latent compression. Compared to a standard dense 3D grid at resolution  $N = 64$  requiring  $64^3$**   
 191 **elements, our structured voxel latent mostly utilize a sparse set of approximately  $L \approx 8000$  active**  
 192 **voxels, each carrying a compact feature  $d = 8$  in our settings. This is similar to the design of 2D**  
 193 **Latent Diffusion Models, where the latent space preserves spatial topology ( $H \times W$ ) but compresses**  
 194 **the channel dimension for efficient generative modeling. This latent representation is connected to**  
 195 **2D multi-view observations with an encoder-decoder framework.**

196 **Encoding from images to 3D latent.** To construct the latent  $z_t$  from multi-view images, a pre-  
 197 trained DINOv2 encoder extracts patch-level embeddings. Then these 2D embeddings are unpro-  
 198 jected in the 3D voxel grid. **For each voxel, the unprojected DINOv2 features from multi-view**  
 199 **images will be averaged to an embedding, then a sparse encoder  $\mathcal{E}$  to produce the latent features  $f_i$ .**

200 **Decoding from 3D latent to images.** To get back to a renderable scene, a sparse decoder  $\mathcal{D}$  maps  
 201 each latent voxel feature  $f_i$  to a set of  $K$  3D Gaussians  $\{(o_i^k, c_i^k, s_i^k, \alpha_i^k, r_i^k)\}_{k=1}^K$ , which can be  
 202 rendered into images from arbitrary viewpoints or be converted into a point cloud. This establishes  
 203 a mapping from the 3D latent  $z_t$  to observation  $o_t^{(i)}$ .

204 **We use the pre-trained 3D encoder-decoder from TRELLIS, which was trained using RGB recon-  
 205 struction losses (L1, D-SSIM, LPIPS) to supervise the 3D Gaussians.** This encoder-decoder archi-  
 206 tecture bridges raw visual perception (2D images) and a structured internal 3D world state ( $z_t$ ). With  
 207 this representation in place, the next step is to learn temporal dynamics in latent space.

## 210 4 4D LATENT WORLD MODEL

212 We propose a *4D latent world model* to predict the dynamics of 3D scenes. The model generates  
 213 future 3D structures conditioned on current observations and textual instructions. With the ability  
 214 to model 3D dynamics, it can serve as a planner for robot manipulation tasks, and when combined  
 215 with an inverse dynamics module, it converts predicted 3D futures into executable robot actions. An  
 216 overview of the framework is shown in Fig. 2.

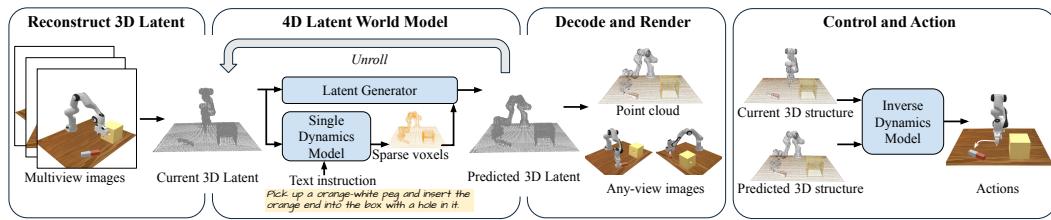


Figure 2: **4D latent world model for robot planning.** The model reconstructs a 3D latent from multi-view images. A 4D latent world model then predicts future latents conditioned on the current state and a text instruction, using a Single Dynamics Model for coarse structural changes and a Latent Generator for detailed features. The predicted latents are decoded into explicit 3D formats such as point clouds or rendered views, which are subsequently used by a goal-conditioned inverse dynamics model to produce robot actions.

#### 4.1 CONDITIONED 3D LATENT SEQUENCE GENERATION

We formulate 4D world modeling as a conditional generator in latent space:  $g(z_{t+1}, \dots, z_{t+T} | z_t, a)$ , as detailed in Section 3. The generator operates autoregressively  $g(z_{t+1} | z_t, a)$ , and future states are obtained by iterative rollout. Due to the complexity of generating a full 3D latent state at once, we adopt a two-step pipeline: a Single Dynamics Model  $SD$  that forecasts the coarse geometry of the next state, and a Latent Generator  $LG$  that fills in detailed feature representations. Together, they construct the next latent  $z_{t+1}$ , which is then fed forward for rollout.

**Data Preparation.** Each training sequence is represented as  $(z_1, \dots, z_T, a)$ . For a robot task, we uniformly sample  $T$  intermediate timesteps as subgoals. At each  $t$ , multi-view images are processed by a pre-trained encoder (Xiang et al., 2025) to obtain a 3D latent  $z_t$ . During training, we randomly choose  $t \in \{1, \dots, T-1\}$  and use  $(z_t, z_{t+1}, a)$  pairs for supervision.

##### 4.1.1 SINGLE DYNAMICS MODEL

The single dynamics model  $SD(\{p_i\}_{t+1} | z_t, a)$  focuses on the dynamics, which predicts the sparse voxels of the next state conditioned on the current latent and the text instruction.

**Modeling.** We use conditional flow matching (Lipman et al., 2022) for generative modeling, which is closely related and largely equivalent in formulation to standard diffusion/score-matching objectives. Here, we adopt flow matching for simplicity and consistency in our setup. The voxel grid  $\{0, 1\}^{N^3}$  is first encoded by 3D convolutional blocks and compressed into a latent tensor  $\mathbb{R}^{N_c^3}$  with lower resolution ( $N_c < N$ ). A transformer denoiser then operates in this compressed space.

**Conditioning.** Text instructions are encoded with a pre-trained CLIP model (Radford et al., 2021). The current latent  $z_t$  is processed by 3D convolutions and aligned to resolution  $N_c$ . Both conditions share positional encodings with the voxel tokens, enabling the model to capture correlations, and are injected in each transformer block through cross-attention. To improve robustness to partial observations, we use condition augmentation, randomly dropping out voxel features from the input latent condition  $z_t$  and adding Gaussian noise to its features  $\{f_i\}$ .

##### 4.1.2 LATENT GENERATOR

The latent generator  $LG(\{f_i\}_{t+1} | \{p_i\}_{t+1}, z_t, a)$  predicts voxel features for the structure given by  $SD$ . Unlike  $SD$ ,  $LG$  focuses on appearance and visual details rather than dynamics. Similar to  $SD$ , it adopts a flow-matching framework with a transformer backbone, conditioned on text and 3D latent features via cross-attention. With this design,  $SD$  and  $LG$  can be trained separately but applied iteratively:  $SD$  predicts voxel positions, and  $LG$  fills in their features, producing complete 3D latents  $z_{t+1}, \dots, z_{t+T}$  over time.

#### 4.2 PLANNING WITH INVERSE DYNAMICS

The 4D latent world model serves as the core of a robotic planner. Given a text instruction  $a$  and the current state latent  $z_0$ , the model predicts future states  $z_1, \dots, z_T$  describing how the agent will interact with the environment.

**Goal-Conditioned Inverse Dynamics.** To translate predicted latents into robot control, we use a goal-conditioned inverse dynamics module:  $ID(s_1, \dots, s_H | z_t, z_{t+1})$ , which outputs an action se-

270 quence  $(s_1, \dots, s_H)$ , **absolute joint positions representing low-level robot commands**, from the current state  $z_t$  to the subgoal  $z_{t+1}$ . Since this model does not perform long-horizon planning, it does  
 271 not require the full details of the latent. Instead, we decode  $z_t$  and  $z_{t+1}$  into lighter point cloud  
 272 representations  $pc_t$  and  $pc_{t+1}$ . Then, a pyramid convolutional encoder (Ze et al., 2024a) to extract  
 273 features from the point clouds, which are concatenated with robot joint states and passed to a diffusion  
 274 head to predict the action sequence. To support different horizons, we randomly vary the action  
 275 length during training and truncate or repeat them to a fixed horizon  $H$ . The inverse dynamics  
 276 module is trained independently of the world model.  
 277

278 **Planning Pipeline.** The complete planning process is summarized in Algorithm 1. Starting from  
 279 initial observations, the world model generates subgoals  $z_1, \dots, z_T$ . The inverse dynamics model  
 280 then predicts action sequences to reach each subgoal  $z_{t+1}$ , repeatedly replanning as needed. For  
 281 closed-loop planning, latents can be updated from new observations after executing actions. This  
 282 integration enables the 4D latent world model to serve as a planner for diverse robotic tasks.

---

283 **Algorithm 1** 4D Latent World Model for Robot Planning

---

285 1: Observe initial multi-view images  $\{o_0^{(i)}\}$ .  
 286 2: Encode initial state:  $z_0 \leftarrow \mathcal{E}(\{o_0^{(i)}\})$ .  
 287 3: Generate future 3D latents  $\{z_1, \dots, z_T\}$  conditioned on  $z_0$  and instruction  $a$ .  
 288 4: **for**  $t = 0, \dots, T - 1$  **do**  
 289 5:   **while** agent has not reached subgoal  $z_{t+1}$  **do**  
 290 6:     Decode latent to point cloud  $pc_t \leftarrow \mathcal{D}(z_t)$  and  $pc_{t+1} \leftarrow \mathcal{D}(z_{t+1})$ .  
 291 7:     Predict action chunk with inverse dynamics model  $s_1, \dots, s_H \leftarrow ID(pc_t, pc_{t+1})$ .  
 292 8:     Agent execute  $H_a \leq H$  actions  $s_1, \dots, s_{H_a}$ .  
 293 9:     **if** close loop planning **then**  
 294 10:       Observe new multi-view images  $\{o_{t+1}^{(i)}\}$ .  
 295 11:       Update next state:  $z_{t+2} \leftarrow LG(SD(\mathcal{E}(o_{t+1}^{(i)})))$   
 296 12:     **end if**  
 297 13:     **end while**  
 298 14: **end for**

---

300 **5 EXPERIMENTS**

302 We evaluate the proposed 4D latent world model on both generation quality and downstream robot  
 303 planning. Our experiments are designed to answer the following key questions:

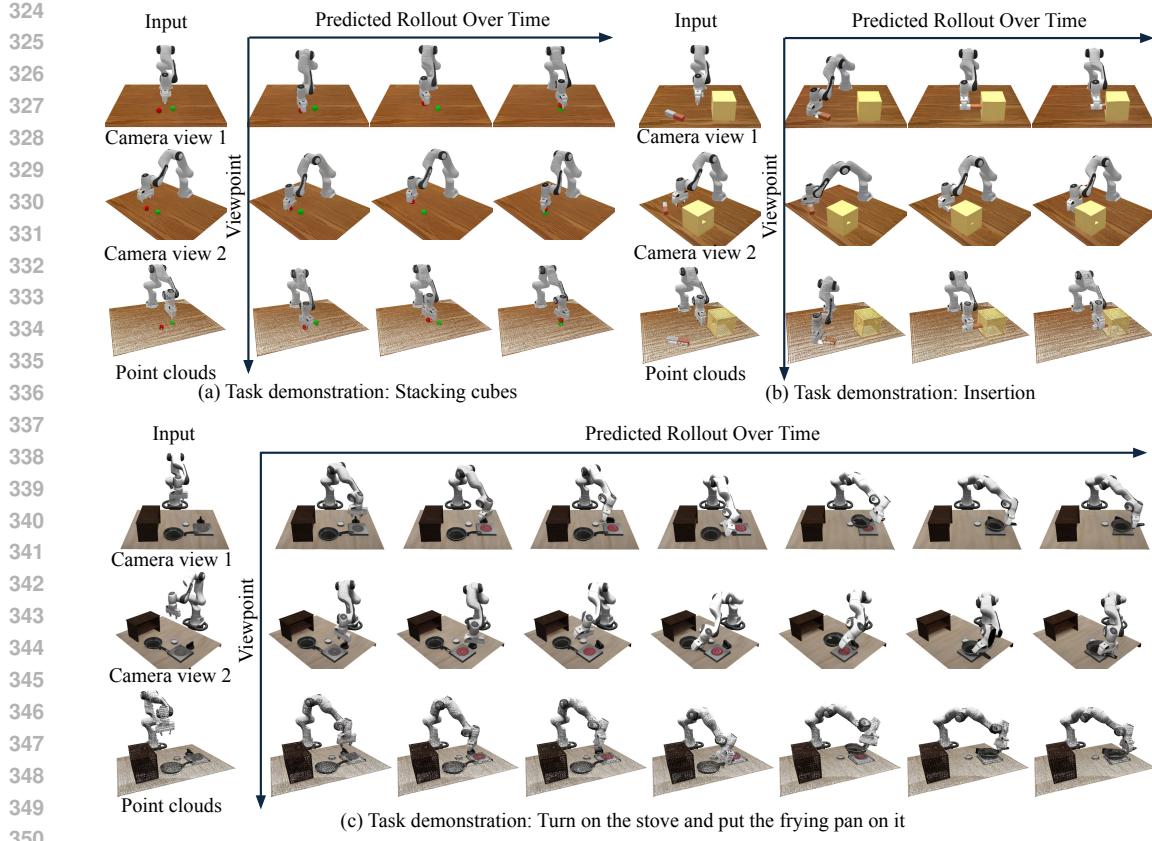
304

- **4D Generation Quality:** How well does our model generate future 3D structures compared to  
 305 state-of-the-art video-based and 4D world models, in terms of visual quality, physical consis-  
 306 tency, and viewpoint invariance?
- **Robot Planning Performance:** Can the generated 3D structures be effectively used for robot  
 307 planning, and how does our approach perform on complex manipulation tasks compared to  
 308 baseline methods?

311 **5.1 EXPERIMENTAL SETUP**

313 **Training Data for 4D Latent World Model.** We collect training data from various robot planning  
 314 tasks in ManiSkill3 (Tao et al., 2025) and LIBERO (Liu et al., 2023) simulators. Each task is paired  
 315 with a language instruction and a set of successful trajectories. For ManiSkill3, we generate 1,000  
 316 demonstrations per task, and for LIBERO-90, 50 demonstrations per task. From each trajectory,  
 317 we uniformly sample 4–10 intermediate timesteps and render multi-view observations using 40  
 318 cameras distributed spherically around the scene. To focus on relevant regions, we remove the  
 319 background outside a pre-defined task-specific bounding box. Following the data preparation  
 320 pipeline described in Section 4.1, each demonstration is converted into a standardized format for  
 321 training the world model.

322 **Inverse Dynamics for Robot Planning.** We evaluate robot planning on three ManiSkill3 tasks:  
 323 StackCube-v1, PullCubeTool-v1, and PegInsertionSide-v1 (Tao et al., 2025). For each task, we col-  
 324 lect 1,000 demonstration trajectories with point cloud observations aggregated from four cameras,



**Figure 3: 4D generation visualizations.** Given input observations in the first column, our model unrolls the 4D latent world to generate future 3D structures over time. The first two rows show renderings from different camera viewpoints, and the third row shows corresponding point cloud visualizations. Text instructions for each task: (a) Pick up a red cube and stack it on top of a green cube, and let go of the cube without it falling. (b) Pick up an orange-white peg and insert the orange end into the box with a hole in it. (c) Turn on the stove and put the frying pan on it.

paired with corresponding action sequences. These demonstrations are used to train the inverse dynamics module (Section 4.2), which converts generated subgoals into executable actions. Evaluation is performed on the same tasks under different random initial conditions. The three tasks require varying levels of 3D understanding: StackCube-v1 involves stacking one cube on another, PullCubeTool-v1 requires using an L-shaped tool to pull a distant cube, and PegInsertionSide-v1 demands precise 3D alignment to insert a peg into a hole. The latter is especially sensitive to fine geometric accuracy; for this task, we relax the success clearance from 0.003 to 0.01.

**Baselines.** We compare our 4D generation and robot planning ability with the following baselines:

- **UniPi** (Du et al., 2023b) is a video planner that generates a single video about the robot manipulation trajectory and leverages inverse dynamics to get the robot control signal. Here we finetune a **Wan 2.1** (Wan et al., 2025) video generation model as the video planner.
- **TesserAct** (Zhen et al., 2025) is a 4D embodied world model that generates a sequence of paired RGB, depth, and normal, which enables robot manipulation.
- **OpenSora** (Zheng et al., 2024) is an image-to-video generation model, which is regarded as a baseline in world modeling generation comparisons.
- **Diffusion Policy** (Chi et al., 2023) and **3D Diffusion Policy** (Ze et al., 2024b) are state-of-the-art imitation learning methods.

UniPi and TesserAct are first finetuned on the same dataset to generate the video trajectory for each task. Then, an image-based diffusion inverse dynamics module is used to convert the generated video plan to robot actions. DP and DP3 are trained on expert demos for each task.

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  
Table 1: **Evaluation of 4D generation.** We collect 5 key frames per trajectory and 40 camera views per frame, and evaluate both standard image quality metrics (PSNR, SSIM, LPIPS) and 3D consistency metrics from MVG Bench (Xie et al., 2025b) (Chamfer Distance, depth error, cPSNR, cSSIM, cLPIPS). Compared to Wan-2.1, TesserAct, and OpenSora, our method achieves the best results on most metrics, with especially large improvements in 3D consistency, owing to its explicit 3D latent representation.

	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	CD $\downarrow$	depth $\downarrow$	cPSNR $\uparrow$	cSSIM $\uparrow$	cLPIPS $\downarrow$
Wan-2.1	19.87	0.84	0.09	43.09	25.06	16.86	0.62	0.24
TesserAct	21.63	<b>0.86</b>	<b>0.07</b>	42.79	23.87	17.91	0.65	0.23
<b>OpenSora-1.3</b>	<b>19.89</b>	<b>0.82</b>	<b>0.09</b>	<b>44.07</b>	<b>25.82</b>	<b>16.67</b>	<b>0.60</b>	<b>0.25</b>
Ours	<b>22.45</b>	0.79	0.13	<b>5.95</b>	<b>9.38</b>	<b>27.42</b>	<b>0.86</b>	<b>0.07</b>

Figure 4: **Novel view generalization.** All models were trained on fixed global views but tested on a novel local viewpoint. Our model generates a consistent 3D scene from an unseen view, outperforming baselines significantly.

## 5.2 4D GENERATION RESULTS

**Visual Quality.** We begin by demonstrating the 4D generation capabilities of our proposed latent world model. Given multi-view images of the initial frame as input, our model autoregressively generates a sequence of future 3D latents to simulate the task’s completion. Figure 3 visualizes the generated rollouts for several tasks. For each trajectory, we render the predicted 3D latents as images from two camera views and as a global point cloud. The results demonstrate that our generated 3D sequences maintain physical plausibility and temporal consistency while exhibiting high visual fidelity.

**Multiview Consistency.** Video generation-based approaches UniPi, TesserAct, and OpenSora, struggle to effectively integrate multi-view information. A common strategy for these models is to generate independent video sequences for each viewpoint and then attempt to fuse them at each timestep. However, without explicit 3D constraints, the independently generated views tend to lose consistency over time, which hinders the ability to leverage this multi-view information for downstream tasks, such as robot planning. In contrast, our model directly generates a unified 3D latent representation, which inherently enforces a consistent 3D structure and thus guarantees multi-view consistency by design. Table 1 presents a quantitative comparison against fine-tuned Wan 2.1, TesserAct, and OpenSora 1.3. As shown in the table, our method significantly outperforms the video-based approaches for multiview consistency.

**Viewpoint Generalization.** Many real-world tasks, particularly in mobile manipulation, cannot rely on fixed sensors and require observations from varying viewpoints. In such scenarios, it is crucial for a world model to generalize to novel viewpoints when simulating planning trajectories. Figure 4 demonstrates our model’s robust ability to integrate diverse multi-view information and generalize to previously unseen viewpoints.

## 5.3 ROBOT PLANNING RESULTS

We evaluate our proposed 4D latent world model as a task planner, extracting actions at each step using a learned, goal-conditioned inverse dynamics model introduced in Section 4.2. We compare

432 Table 2: **Success rate for robot manipulation tasks.** Average success rate over 100 episodes, using four global  
 433 cameras for observation. For PegInsertionSide-v1, the success clearance is relaxed to 0.01.

	StackCube-v1	PullCubeTool-v1	PegInsertionSide-v1*	Average
DP	56%	87%	<b>24%</b>	55.7%
DP3	47%	<b>94%</b>	7%	49.3%
UniPi	9%	5%	1%	5.0%
TesserAct	13%	1%	3%	5.7%
Ours	<b>84%</b>	84%	16%	<b>61.3%</b>

441 Table 3: **Zero-shot generalization with visual and viewpoint changes.** Success rates on the StackCube-v1  
 442 task under unseen conditions at test-time. Perturbations include reduced lighting, additive Gaussian noise, a  
 443 background color shift (R-channel change for table), and horizontal camera rotations ( $5^\circ, 10^\circ$ ).  
 444

	Lighting	Noise	Background color	Viewpoint (5%)	Viewpoint (10%)
DP	7%	5%	1%	43%	25%
DP3	47%	47%	47%	49%	45%
Ours	<b>78%</b>	<b>80%</b>	<b>84%</b>	<b>85%</b>	<b>83%</b>

450 our method’s manipulation performance against world modeling baselines UniPi and TesserAct,  
 451 and state-of-the-art imitation learning policies DP and DP3. As shown in Table 2, our method  
 452 significantly outperforms the video-based world models and achieves performance comparable to  
 453 the specialized imitation learning policies. It is worth noting that the original DP3 implementation  
 454 does not use color information for better generalization ability, which prevents it from distinguishing  
 455 between colored objects in the StackCube-v1 task. [More robot planning results can be found in](#)  
 456 [Appendix B.1](#).

457 **Zero-shot generalization to visual and viewpoint changes.** Zero-shot generalization to novel vi-  
 458 sual conditions and camera views is critical for deploying robotic policies in real-world, unstructured  
 459 environments. Some recent works (Zhu et al., 2024) have mentioned this point with some studies  
 460 explicitly evaluating robustness to such changes. As demonstrated in Section 5.2, our model uses  
 461 an explicit 3D latent representation, which naturally provides robustness to viewpoint changes. We  
 462 now evaluate the policy’s zero-shot planning performance under various perturbations, including  
 463 changes in lighting, background color, additive image noise, and camera viewpoint. The results in  
 464 Table 3 show that our method maintains a high success rate across these visual changes, demon-  
 465 strating strong zero-shot generalization ability.

#### 467 5.4 ABLATION STUDY

468 **Inputs to the Inverse Dynamics Module.** To understand the design choices of the inverse dynamics  
 469 model, we evaluate three input types: using (i) our default downsampled point cloud (ii) the full 3D  
 470 latents, and (iii) the 3D voxels as input to the inverse dynamic module. Due to the heavy computation  
 471 complexity for 3D latents encoding with large number of camera views, here we use 4 cameras for  
 472 inverse dynamics module training. Table 4 shows the success rate for robot task, which demonstrates  
 473 the downsampled point cloud achieves performance comparable to latent-based inputs, providing a  
 474 strong and lightweight signal for predicting robot actions.

475 **Number of camera views.** We train world models with 4, 10, and 40 camera views while keeping  
 476 inference to 4 views. Planning success and 3D consistency improve with additional training views  
 477 (Table 5 and Table 6), but even the 4-view model substantially outperforms video-based baselines.  
 478 This shows the method remains effective under limited multi-view supervision.

#### 479 5.5 REAL WORLD EXPERIMENTS

480 To evaluate the real-world applicability of our model, we collected a dataset of 100 human demon-  
 481 strations for a physical block-in-basket task using five RGB cameras. From each trajectory, we  
 482 uniformly sampled five intermediate frames and encoded them into 3D latents representation using  
 483 the pre-trained encoder to form the training set. Figure 5 (a) illustrates our data collection setup.

486 Table 4: **Ablation on inputs to the inverse dynamics module: success rate on the StackCube-v1 task.**

Point cloud (40 cams)	Point cloud (4 cams)	3D Latents (4 cams)	3D voxels (4 cams)
84 %	57 %	59%	40%

490 Table 5: **Ablation on the number of training-time camera views: visual consistency metrics on the StackCube-491 v1 task. All models use 4 cameras at inference time.**

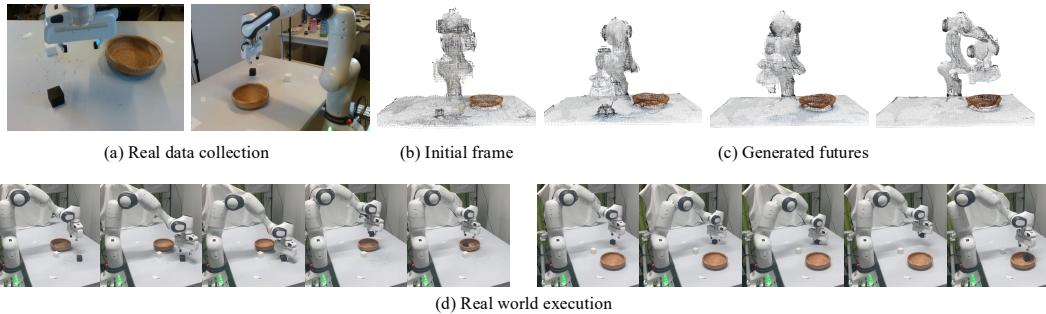
	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	CD $\downarrow$	depth $\downarrow$	cPSNR $\uparrow$	cSSIM $\uparrow$	cLPIPS $\downarrow$
Wan-2.1	20.10	0.84	0.09	38.74	24.54	16.95	0.62	0.22
TesserAct	22.26	<b>0.87</b>	<b>0.06</b>	39.11	24.86	17.75	0.64	0.22
Ours (4 cams)	19.81	0.75	0.18	7.10	<b>8.81</b>	<b>28.89</b>	<b>0.86</b>	0.07
Ours (10 cams)	21.78	0.77	0.14	7.06	9.85	26.92	0.85	0.07
Ours (40 cams)	<b>22.39</b>	0.78	0.13	<b>6.81</b>	9.98	26.75	0.85	<b>0.07</b>

492 We trained our 4D latent world model and inverse dynamics module on the collected real-world  
 493 dataset, using the same configuration as in our simulation experiments. Figure 5 (b) and (c) illustrate  
 494 qualitative generation results for real-world scenarios, while Figure 5 (d) presents visualizations of  
 495 two policy rollouts. The generated point cloud sequences exhibit temporal and physical consistency,  
 496 and the successful demonstrations indicate that our model learns meaningful dynamics from real-  
 497 world data.

498 To quantitatively compare our approach with baselines, we randomly initialized object positions  
 499 and evaluated our method against the Diffusion Policy (DP) over 50 episodes. Our method achieved  
 500 a success rate comparable to DP (Ours 52%, DP 50%), demonstrating that our proposed model  
 501 performs effectively in real-world robotic manipulation settings.

502 Table 6: **Ablation on the number of training-time camera views: planning success rate on the StackCube-v1  
 503 task. The number of camera views refers to the world-model training setup; inference always uses 4 cameras.**

Ours (40 cams)	Ours (10 cams)	Ours (4 cams)	DP	DP3	UniPi	TesserAct
84%	72%	57%	56%	47%	9%	13%

526 Figure 5: **Real world experiments.** We collect real robot data (a), reconstruct the initial input frame from these  
 527 observations (b), predict future rollouts in real environment (c), and execute proposed policy at test time (d).

## 529 6 CONCLUSION

530 We have introduced a *4D latent world model* for robot planning, which predicts the evolution of 3D  
 531 scene structure directly in a compact latent space. By moving beyond prevailing 2D video-based  
 532 approaches, our model learns a dynamic model in 3D latent space that encodes holistic scene struc-  
 533 ture to enforce 3D consistency, producing rollouts of future latents that can be decoded into explicit  
 534 formats such as point clouds or rendered views. Integrated with a goal-conditioned inverse dyna-  
 535 mics module, these latents serve as geometrically grounded subgoals that translate into executable  
 536 actions. Our experiments demonstrate that this approach achieves state-of-the-art performance in  
 537 3D-aware generative modeling, yielding significant improvements in downstream robotic planning  
 538 tasks. While our current implementation assumes calibrated multi-view inputs to reconstruct the  
 539 initial latent, extending to weaker input settings is a promising direction for broader applicability.

540  
541  
**ETHICS STATEMENT**542  
543  
544  
545  
546  
The authors have considered the ethical implications of this research and have found no direct ethical  
concerns. Our work is foundational, focusing on improving the planning capabilities of robotic  
agents. The data used for training and evaluation is sourced from established public robotics benchmarks,  
and our small-scale real-world data collection did not involve sensitive information or raise  
privacy concerns. We believe our work adheres to the ICLR Code of Ethics.547  
548  
**REPRODUCIBILITY STATEMENT**  
549550  
551  
552  
553  
554  
We are committed to ensuring the reproducibility of our work. We plan to release our full imple-  
mentation after a thorough code cleanup and documentation process. All simulated data used in our  
experiments was generated using the official, publicly available APIs of the ManiSkill and LIBERO  
benchmark suites. To ensure fair and robust comparisons, all reported metrics and success rates  
were obtained using fixed random seeds and consistent evaluation environments.555  
556  
**REFERENCES**  
557558  
559  
Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal.  
Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022. 3  
560  
561  
Anurag Ajay, Seungwook Han, Yilun Du, Shuang Li, Abhi Gupta, Tommi Jaakkola, Josh Tenen-  
baum, Leslie Kaelbling, Akash Srivastava, and Pulkit Agrawal. Compositional foundation models  
for hierarchical planning. *Advances in Neural Information Processing Systems*, 36:22304–22325,  
2023. 3  
562  
563  
564  
565  
Eloi Alonso, Adam Jolley, Vincent Micheli, Anssi Kanervisto, Amos J Storkey, Tim Pearce, and  
Fran ois Fleuret. Diffusion for world modeling: Visual details matter in atari. *Advances in Neural Information Processing Systems*, 37:58757–58791, 2024. 3  
566  
567  
568  
569  
Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models.  
In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15791–15801,  
2025. 3  
570  
571  
572  
573  
574  
575  
576  
Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-  
Hsin Fang, Kuniyatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, et al. A care-  
ful examination of large behavior models for multitask dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025. 2  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolò  
Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al.  $\pi_0$ : A vision-language-action flow  
model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. 2  
594  
595  
596  
597  
598  
599  
Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris,  
Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. Sam 3: Segment  
anything with concepts. *arXiv preprint arXiv:2511.16719*, 2025. 16  
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  
7010  
7011  
7012  
7013  
7014  
7015  
7016  
7017  
7018  
7019  
7020  
7021  
7022  
7023  
7024  
7025  
7026  
7027  
7028  
7029  
7030  
7031  
7032  
7033  
7034  
7035  
7036  
7037  
7038  
7039  
7040  
7041  
7042  
7043  
7044  
7045  
7046  
7047  
7048  
7049  
7050  
7051  
7052  
7053  
7054  
7055  
7056  
7057  
7058  
7059  
7060  
7061  
7062  
7063  
7064  
7065  
7066  
7067  
7068  
7069  
7070  
7071  
7072  
7073  
7074  
7075  
7076  
7077  
7078  
7079  
7080  
7081  
7082  
7083  
7084  
7085  
7086  
7087  
7088  
7089  
7090  
7091  
7092  
7093  
7094  
7095  
7096  
7097  
7098  
7099  
70100  
70101  
70102  
70103  
70104  
70105  
70106  
70107  
70108  
70109  
70110  
70111  
70112  
70113  
70114  
70115  
70116  
70117  
70118  
70119  
70120  
70121  
70122  
70123  
70124  
70125  
70126  
70127  
70128  
70129  
70130  
70131  
70132  
70133  
70134  
70135  
70136  
70137  
70138  
70139  
70140  
70141  
70142  
70143  
70144  
70145  
70146  
70147  
70148  
70149  
70150  
70151  
70152  
70153  
70154  
70155  
70156  
70157  
70158  
70159  
70160  
70161  
70162  
70163  
70164  
70165  
70166  
70167  
70168  
70169  
70170  
70171  
70172  
70173  
70174  
70175  
70176  
70177  
70178  
70179  
70180  
70181  
70182  
70183  
70184  
70185  
70186  
70187  
70188  
70189  
70190  
70191  
70192  
70193  
70194  
70195  
70196  
70197  
70198  
70199  
70200  
70201  
70202  
70203  
70204  
70205  
70206  
70207  
70208  
70209  
70210  
70211  
70212  
70213  
70214  
70215  
70216  
70217  
70218  
70219  
70220  
70221  
70222  
70223  
70224  
70225  
70226  
70227  
70228  
70229  
70230  
70231  
70232  
70233  
70234  
70235  
70236  
70237  
70238  
70239  
70240  
70241  
70242  
70243  
70244  
70245  
70246  
70247  
70248  
70249  
70250  
70251  
70252  
70253  
70254  
70255  
70256  
70257  
70258  
70259  
70260  
70261  
70262  
70263  
70264  
70265  
70266  
70267  
70268  
70269  
70270  
70271  
70272  
70273  
70274  
70275  
70276  
70277  
70278  
70279  
70280  
70281  
70282  
70283  
70284  
70285  
70286  
70287  
70288  
70289  
70290  
70291  
70292  
70293  
70294  
70295  
70296  
70297  
70298  
70299  
702100  
702101  
702102  
702103  
702104  
702105  
702106  
702107  
702108  
702109  
702110  
702111  
702112  
702113  
702114  
702115  
702116  
702117  
702118  
702119  
702120  
702121  
702122  
702123  
702124  
702125  
702126  
702127  
702128  
702129  
702130  
702131  
702132  
702133  
702134  
702135  
702136  
702137  
702138  
702139  
702140  
702141  
702142  
702143  
702144  
702145  
702146  
702147  
702148  
702149  
702150  
702151  
702152  
702153  
702154  
702155  
702156  
702157  
702158  
702159  
702160  
702161  
702162  
702163  
702164  
702165  
702166  
702167  
702168  
702169  
702170  
702171  
702172  
702173  
702174  
702175  
702176  
702177  
702178  
702179  
702180  
702181  
702182  
702183  
702184  
702185  
702186  
702187  
702188  
702189  
702190  
702191  
702192  
702193  
702194  
702195  
702196  
702197  
702198  
702199  
702200  
702201  
702202  
702203  
702204  
702205  
702206  
702207  
702208  
702209  
702210  
702211  
702212  
702213  
702214  
702215  
702216  
702217  
702218  
702219  
702220  
702221  
702222  
702223  
702224  
702225  
702226  
702227  
702228  
702229  
702230  
702231  
702232  
702233  
702234  
702235  
702236  
702237  
702238  
702239  
702240  
702241  
702242  
702243  
702244  
702245  
702246  
702247  
702248  
702249  
702250  
702251  
702252  
702253  
702254  
702255  
702256  
702257  
702258  
702259  
702260  
702261  
702262  
702263  
702264  
702265  
702266  
702267  
702268  
702269  
702270  
702271  
702272  
702273  
702274  
702275  
702276  
702277  
702278  
702279  
702280  
702281  
702282  
702283  
702284  
702285  
702286  
702287  
702288  
702289  
702290  
702291  
702292  
702293  
702294  
702295  
702296  
702297  
702298  
702299  
702300  
702301  
702302  
702303  
702304  
702305  
702306  
702307  
702308  
702309  
702310  
702311  
702312  
702313  
702314  
702315  
702316  
702317  
702318  
702319  
702320  
702321  
702322  
702323  
702324  
702325  
702326  
702327  
702328  
702329  
702330  
702331  
702332  
702333  
702334  
702335  
702336  
702337  
702338  
702339  
702340  
702341  
702342  
702343  
702344  
702345  
702346  
702347  
702348  
702349  
702350  
702351  
702352  
702353  
702354  
702355  
702356  
702357  
702358  
702359  
702360  
702361  
702362  
702363  
702364  
702365  
702366  
702367  
702368  
702369  
702370  
702371  
702372  
702373  
702374  
702375  
702376  
702377  
702378  
702379  
702380  
702381  
702382  
702383  
702384  
702385  
702386  
702387  
702388  
702389  
702390  
702391  
702392  
702393  
702394  
702395  
702396  
702397  
702398  
702399  
702400  
702401  
702402  
702403  
702404  
702405  
702406  
702407  
702408  
702409  
702410  
702411  
702412  
702413  
702414  
702415  
702416  
702417  
702418  
702419  
702420  
702421  
702422  
702423  
702424  
702425  
702426  
702427  
702428  
702429  
702430  
702431  
702432  
702433  
702434  
702435  
702436  
702437  
702438  
702439  
702440  
702441  
702442  
702443  
702444  
702445  
702446  
702447  
702448  
702449  
702450  
702451  
702452  
702453  
702454  
702455  
702456  
702457  
702458  
702459  
702460  
702461  
702462  
702463  
702464  
702465  
702466  
702467  
702468  
702469  
702470  
702471  
702472  
702473  
702474  
702475  
702476  
702477  
702478  
702479  
702480  
702481  
702482  
702483  
702484  
702485  
702486  
702487  
702488  
702489  
702490  
702491  
702492  
702493  
702494  
702495  
702496  
702497  
702498  
702499  
702500  
702501  
702502  
702503  
702504  
702505  
702506  
702507  
702508  
702509  
702510  
702511  
702512  
702513  
702514  
702515  
702516  
702517  
702518  
702519  
702520  
702521  
702522  
702523  
702524  
702525  
702526  
702527  
702528  
702529  
702530  
702531  
702532  
702533  
702534  
702535  
702536  
702537  
702538  
702539  
702540  
702541  
702542  
702543  
702544  
702545  
702546  
702547  
702548  
702549  
702550  
702551  
702552  
702553  
702554  
702555  
702556  
702557  
702558  
702559  
702560  
702561  
702562  
702563  
702564  
702565  
702566  
702567  
702568  
702569  
702570  
702571  
702572  
702573  
702574  
702575  
702576  
702577  
702578  
702579  
702580  
702581  
702582  
702583  
702584  
702585  
702586  
702587  
702588  
702589  
702590  
702591  
702592  
702593  
702594  
702595  
702596  
702597  
702598  
702599  
702600  
702601  
702602  
702603  
702604  
702605  
702606  
702607  
702608  
702609  
702610  
702611  
702612  
702613  
702614  
702615  
702616  
702617  
702618  
702619  
702620  
702621  
702622  
702623  
702624  
702625  
702626  
702627  
702628  
702629  
702630  
702631  
702632  
702633  
702634  
702635  
702636  
702637  
702638  
702639  
702640  
702641  
702642  
702643  
702644  
702645  
702646  
702647  
702648  
702649  
702650  
702651  
702652  
702653  
702654  
702655  
702656  
702657  
702658  
702659  
702660  
702661  
702662  
702663  
702664  
702665  
702666  
702667  
702668  
702669  
702670  
702671  
702672  
702673  
702674  
702675  
702676  
702677  
702678  
702679  
702680  
702681  
702682  
702683  
702684  
702685  
702686  
702687  
702688  
702689  
702690  
702691  
702692  
702693  
702694  
702695  
702696  
702697  
702698  
702699  
702700  
702701  
702702  
702703  
702704  
702705  
702706  
702707  
702708  
702709  
702710  
702711  
702712  
702713  
702714  
702715  
702716  
702717  
702718  
702719  
702720  
702721  
702722  
702723  
702724  
702725  
702726  
702727  
702728  
702729  
7027230  
7027231  
7027232  
7027233  
7027234  
7027235  
7027236  
7027237  
7027238  
7027239  
70272310  
70272311  
70272312  
70272313  
70272314  
70272315  
70272316  
70272317  
70272318  
70272319  
70272320  
70272321  
70272322  
70272323  
70272324  
70272325  
70272326  
70272327  
70272328  
70272329  
70272330  
70272331  
70272332  
70272333  
70272334  
70272335  
70272336  
70272337  
70272338  
70272339  
70272340  
70272341  
70272342  
70272343  
70272344  
70272345  
70272346  
70272347  
70272348  
70272349  
70272350  
70272351  
70272352  
70272353  
70272354  
70272355  
70272356  
70272357  
70272358  
70272359  
70272360  
70272361  
70272362  
70272363  
70272364  
70272365  
70272366  
70272367  
70272368  
70272369  
70272370  
70272371  
70272372  
70272373  
70272374  
70272375  
70272376  
70272377  
70272378  
70272379  
70272380  
70272381  
70272382  
70272383  
70272384  
70272385  
70272386  
70272387  
70272388  
70272389  
70272390  
70272391  
70272392  
70272393  
70272394  
70272395  
70272396  
70272397  
70272398  
70272399  
702723100  
702723101  
702723102  
702723103  
702723104  
702723105  
702723106  
702723107  
702723108  
702723109  
702723110  
702723111  
702723112  
702723113  
702723114  
702723115  
702723116  
702723117  
702723118  
702723119  
702723120  
702723121  
702723122  
702723123  
702723124  
702723125  
702723126  
702723127  
702723128  
702723129  
702723130  
702723131  
702723132  
702723133  
702723134  
702723135  
702723136  
702723137  
702723138  
702723139  
702723140  
702723141  
702723142  
702723143  
702723144  
702723145  
702723146  
702723147  
702723148  
702723149  
702723150  
702723151  
702723152  
702723153  
702723154  
702723155  
702723156  
702723157  
702723158  
702723159  
702723160  
702723161  
702723162  
702723163  
702723164  
702723165  
702723166  
702723167  
702723168  
702723169  
702723170  
702723171  
702723172  
702723173  
702723174  
702723175  
702723176  
702723177  
702723178  
702723179  
702723180  
702723181  
702723182  
702723183  
702723184  
702723185  
702723186  
702723187  
702723188  
702723189  
702723190  
702723191  
702723192  
702723193  
702723194  
702723195  
702723196  
702723197  
702723198  
702723199  
702723200  
702723201  
702723202  
702723203  
702723204  
702723205  
702723206  
702723207  
702723208  
702723209  
702723210  
702723211  
702723212  
702723213  
702723214  
702723215  
702723216  
702723217  
702723218  
702723219  
702723220  
702723221  
702723222  
702723223  
702723224  
702723225  
702723226  
702723227  
702723228  
702723229  
702723230  
702723231  
7

594 Danny Driess, Zhiao Huang, Yunzhu Li, Russ Tedrake, and Marc Toussaint. Learning multi-object  
 595 dynamics with compositional neural radiance fields. In *Conference on robot learning*, pp. 1755–  
 596 1768. PMLR, 2023. 3

597

598 Yilun Du, Mengjiao Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet,  
 599 Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, et al. Video language planning. *arXiv preprint*  
 600 *arXiv:2310.10625*, 2023a. 3

601 Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and  
 602 Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural*  
 603 *information processing systems*, 36:9156–9172, 2023b. 1, 3, 7, 15, 16

604

605 Carlos E Garcia, David M Prett, and Manfred Morari. Model predictive control: Theory and prac-  
 606 tice—a survey. *Automatica*, 25(3):335–348, 1989. 1

607

608 Haoran He, Chenjia Bai, Kang Xu, Zhuoran Yang, Weinan Zhang, Dong Wang, Bin Zhao, and Xue-  
 609 long Li. Diffusion model is an effective planner and data synthesizer for multi-task reinforcement  
 610 learning. *Advances in neural information processing systems*, 36:64896–64917, 2023. 3

611

612 Zhi Hou, Tianyi Zhang, Yuwen Xiong, Haonan Duan, Hengjun Pu, Ronglei Tong, Chengyang Zhao,  
 613 Xizhou Zhu, Yu Qiao, Jifeng Dai, and Yuntao Chen. Dita: Scaling diffusion transformer for  
 614 generalist vision-language-action policy. *arXiv preprint arXiv:2503.19757*, 2025. 2

615

616 Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puha Li, Yan Wang, Qing Li,  
 617 Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world.  
 618 *arXiv preprint arXiv:2311.12871*, 2023. 2

619

620 Yixuan Huang, Nichols Crawford Taylor, Adam Conkey, Weiyu Liu, and Tucker Hermans. Latent  
 621 space planning for multiobject manipulation with environment-aware relational classifiers. *IEEE*  
 622 *Transactions on Robotics*, 40:1724–1739, 2024. 3

623

624 Yixuan Huang, Christopher Agia, Jimmy Wu, Tucker Hermans, and Jeannette Bohg. Points2plans:  
 625 From point clouds to long-horizon plans with composable relational dynamics. In *2025 IEEE*  
 626 *International Conference on Robotics and Automation (ICRA)*, pp. 1208–1216. IEEE, 2025. 3

627

628 Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for  
 629 flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022. 3

630

631 Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. Physt-  
 632 win: Physics-informed reconstruction and simulation of deformable objects from videos. *ICCV*,  
 633 2025. 3

634

635 Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion  
 636 with 3d scene representations. *arXiv preprint arXiv:2402.10885*, 2024. 1

637

638 Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splat-  
 639 tiling for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. 2,  
 640 4

641

642 Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair,  
 643 Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source  
 644 vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. 2

645

646 Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to Act  
 647 from Actionless Videos through Dense Correspondences. *arXiv:2310.08576*, 2023. 3

648

649 Kuang-Huei Lee, Ofir Nachum, Mengjiao Yang, Lisa Lee, Daniel Freeman, Winnie Xu, Sergio  
 650 Guadarrama, Ian Fischer, Eric Jang, Henryk Michalewski, et al. Multi-game decision transform-  
 651 ers. *arXiv preprint arXiv:2205.15241*, 2022. 2

652

653 Wenhao Li. Efficient planning with latent diffusion. *arXiv preprint arXiv:2310.00311*, 2023. 3

654

655 Wenhao Li, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Hierarchical diffusion for offline decision  
 656 making. In *International Conference on Machine Learning*, pp. 20035–20064. PMLR, 2023. 3

648 Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa,  
 649 David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv*  
 650 *preprint arXiv:1509.02971*, 2015. 1

651 Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching  
 652 for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 5

653 Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero:  
 654 Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*,  
 655 2023. 6

656 David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained  
 657 model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000. 1

658 Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and  
 659 Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications*  
 660 *of the ACM*, 65(1):99–106, 2021. 2, 4

661 NVIDIA, Nikita Cherniadev, Johan Bjorck, Fernando Castañeda, Xingye Da, Runyu Ding,  
 662 Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang,  
 663 Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu,  
 664 Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed,  
 665 You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen  
 666 Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng,  
 667 and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots. In *ArXiv*  
 668 *Preprint*, March 2025. 2

669 Han Qi, Haocheng Yin, Aris Zhu, Yilun Du, and Heng Yang. Strengthening generative robot policies  
 670 through predictive world modeling. *arXiv preprint arXiv:2502.00622*, 2025. 1, 3

671 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal,  
 672 Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual  
 673 models from natural language supervision. In *International conference on machine learning*, pp.  
 674 8748–8763. PMLR, 2021. 5

675 Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov,  
 676 Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al.  
 677 A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022. 2, 3

678 Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-  
 679 resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF confer-  
 680 ence on computer vision and pattern recognition*, pp. 10684–10695, 2022. 2

681 Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see,  
 682 simulate, and shape elasto-plastic objects with graph networks. *arXiv preprint arXiv:2205.02909*,  
 683 2022. 3

684 Haochen Shi, Huazhe Xu, Samuel Clarke, Yunzhu Li, and Jiajun Wu. Robocook: Long-horizon  
 685 elasto-plastic object manipulation with diverse tools. *arXiv preprint arXiv:2306.14447*, 2023. 3

686 Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao,  
 687 Xinsong Lin, Yulin Liu, Tse kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav  
 688 Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Ca-  
 689 landra, Rui Chen, Shan Luo, and Hao Su. Maniskill3: Gpu parallelized robotics simulation and  
 690 rendering for generalizable embodied ai. *Robotics: Science and Systems*, 2025. 6

691 Toshihide Ubukata, Jialong Li, and Kenji Tei. Diffusion model for planning: A systematic literature  
 692 review. *arXiv preprint arXiv:2408.10266*, 2024. 3

693 Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu,  
 694 Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai  
 695 Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi  
 696 Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang,  
 697

702 Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng  
 703 Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan  
 704 Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You  
 705 Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen  
 706 Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models.  
 707 *arXiv preprint arXiv:2503.20314*, 2025. 7, 15, 16

708 Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen,  
 709 Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation.  
 710 In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 21469–21480,  
 711 2025. 2, 4, 5, 15

712 Amber Xie, Oleh Rybkin, Dorsa Sadigh, and Chelsea Finn. Latent diffusion planning for imitation  
 713 learning. *arXiv preprint arXiv:2504.16925*, 2025a. 3

714 Xianghui Xie, Chuhang Zou, Meher Gitika Karumuri, Jan Eric Lenssen, and Gerard Pons-Moll.  
 715 Mvgbench: Comprehensive benchmark for multi-view generation models, 2025b. 8

716 Haoyu Xiong, Quanzhou Li, Yun-Chun Chen, Homanga Bharadhwaj, Samarth Sinha, and Animesh  
 717 Garg. Learning by watching: Physical imitation of manipulation skills from human videos. In  
 718 *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7827–  
 719 7834, 2021. doi: 10.1109/IROS51168.2021.9636080. 1

720 Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter  
 721 Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 1(2):6,  
 722 2023. 1, 3

723 Yanjie Ze, Zixuan Chen, Wenhao Wang, Tianyi Chen, Xialin He, Ying Yuan, Xue Bin Peng, and  
 724 Jiajun Wu. Generalizable humanoid manipulation with 3d diffusion policies. *arXiv preprint  
 725 arXiv:2410.10803*, 2024a. 6, 15

726 Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion  
 727 policy: Generalizable visuomotor policy learning via simple 3d representations. In *Proceedings  
 728 of Robotics: Science and Systems (RSS)*, 2024b. 7

729 Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual  
 730 Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*,  
 731 Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016. 1

732 Haoyu Zhen, Qiao Sun, Hongxin Zhang, Junyan Li, Siyuan Zhou, Yilun Du, and Chuang Gan.  
 733 Tesseract: learning 4d embodied world models. *arXiv preprint arXiv:2504.20995*, 2025. 2, 3, 7,  
 734 15, 16

735 Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun  
 736 Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all.  
 737 *arXiv preprint arXiv:2412.20404*, 2024. 7, 16

738 Haoyi Zhu, Yating Wang, Di Huang, Weicai Ye, Wanli Ouyang, and Tong He. Point cloud matters:  
 739 Rethinking the impact of different observation spaces on robot learning. *Advances in Neural  
 740 Information Processing Systems*, 37:77799–77830, 2024. 1, 9

741 Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart,  
 742 Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge  
 743 to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023. 2

744

745

746

747

748

749

750

751

752

753

754

755

756 THE USE OF LARGE LANGUAGE MODELS  
757758 Large Language Models (LLMs) were used to assist with manuscript language polishing and litera-  
759 ture search. All substantive research content and scientific conclusions are the original work of the  
760 authors.  
761762 A IMPLEMENTATION DETAILS  
763764 A.1 4D WORLD MODEL TRAINING  
765766 Our framework consists of two main components: a single dynamics model and a latent generator,  
767 which were trained independently. Both were implemented as conditioned flow matching models  
768 following the architecture proposed by Xiang et al. (2025). Here, we extend the original conditions  
769 to both text and 3D latent. The latent generator operates on a  $64 \times 64 \times 64$  voxelized grid with  
770 a feature dimension of  $d = 8$ , while the dynamics model uses a similar architecture on a coarser  
771  $16 \times 16 \times 16$  grid. Both models consist of 24 transformer blocks, each with 16 attention heads  
772 and a model dimension of 1024. For text conditioning, we utilized embeddings from a pre-trained  
773 CLIP model. The dynamics model is also conditioned on the input 3D latent; we use sparse 3D  
774 convolutional layers to match the latent’s resolution to the model’s internal dimension, after which  
775 it is injected into the cross-attention blocks together with the text condition.  
776777 Training for both models spanned 300,000 steps with a learning rate of  $1 \times 10^{-4}$ . We used a per-  
778 GPU batch size of 8 with mixed-precision (FP16) computation. For the latent generator, we applied  
779 4 gradient accumulation steps, and for the dynamics model, we used 2 steps. The optimizer used  
780 was AdamW with no weight decay. To enable classifier-free guidance, we set the unconditional  
781 dropout probability to 0.1 and applied an exponential moving average (EMA) with a decay rate of  
782 0.9999 to stabilize the training. Each model was trained for approximately 3 days on four NVIDIA  
783 H100 (80GB) GPUs.  
784785 A.2 INVERSE DYNAMICS MODEL TRAINING  
786787 Our goal-conditioned inverse dynamics model is trained on 1,000 expert demonstrations for each  
788 task. The policy is formulated as a diffusion model that takes point clouds representing the current  
789 and goal states as input. For observation encoding, we employ a point cloud encoder introduced by  
790 Ze et al. (2024a), which comprises four 1D convolutional layers with a hidden dimension of 128.  
791 The action decoder is a 1D conditional UNet with downsampling channel dimensions of [256, 512,  
792 1024], a kernel size of 5, and 8 groups for normalization layers.  
793794 The inverse dynamics model was trained for 20,000 epochs. At inference time, we use 100 denoising  
795 steps to predict the action sequence. Training for each task took approximately 8 hours on a single  
796 NVIDIA A100 GPU.  
797798 A.3 VIDEO WORLD MODEL BASELINE FINE-TUNING  
799800 To establish our baselines, we utilize Wan 2.1 (Wan et al., 2025) as the video generative backbone  
801 of UniPi (Du et al., 2023b), and the TesserAct (Zhen et al., 2025) generative model. Both models  
802 were fine-tuned on the same dataset. For the TesserAct model specifically, we generated depth  
803 maps alongside the images from the simulator and used its official codebase to create normal maps.  
804 For fine-tuning, Wan 2.1 was trained for 10,000 steps on two NVIDIA H100 GPUs, achieving  
805 convergence in approximately 36 hours. TesserAct was also fine-tuned for 10,000 steps on four  
806 NVIDIA H100 GPUs, taking around two days to converge.  
807808 B ADDITIONAL EXPERIMENT RESULTS  
809810 B.1 ROBOT PLANNING RESULTS  
811812 In Section 5.3, we compare our proposed methods with baselines TesserAct and UniPi. However, the  
813 success rate of both methods in ManiSkill3 tasks are very low. To provide a more rigorous evalua-  
814

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  
 tion of robot planning capabilities, we extended our experiments to the RLBench benchmark, where TesserAct (Zhen et al., 2025) and UniPi (Du et al., 2023b) have established performance records. We evaluated our method on three RLBench tasks: *CloseBox*, *SweepToDustpan*, and *WaterPlants*, collecting 1,000 demonstrations for each. We utilized 20 cameras for model training and 4 cameras for inference, maintaining the same 4D world model and inverse dynamics architecture described in Section 5.3. For the baselines, we cite the success rates reported in the official TesserAct publication (Zhen et al., 2025). For our method, we report the success rate averaged over 100 random episodes per task. Table 7 shows the success rate comparison in 3 RLBench tasks.

Table 7: **Success rate on RLBench.** Average success rate over 100 episodes for our model. For Image-BC, UniPi, and TesserAct baselines, the success rate is from Zhen et al. (2025).

	Close Box	Sweep To Dustpan	Water Plants	Average
Image-BC	53%	0%	0%	17.7%
UniPi	81%	49%	35%	55.0%
TesserAct	88%	56%	41%	61.7%
Ours	<b>93%</b>	<b>69%</b>	<b>64%</b>	<b>75.3%</b>

## B.2 4D GENERATION RESULTS

In Section 5.2, we compare the performance of 4D generation results with video generation based baselines. We have shown that our proposed model performs a strong ability in multiview consistency and viewpoint generalization, while maintaining the comparable visual quality at the same time. In order to further evaluate the connection of world modeling quality and robot policy performance more directly, we use Segment Anything Model 3 (SAM3) (Carion et al., 2025) to segment the robot shape in each generated image, and compare the IoU score of robot mask between generation and ground truth. Table 8 shows the IoU score comparison with OpenSora-1.3 (Zheng et al., 2024), Wan-2.1 (Wan et al., 2025), and TesserAct (Zhen et al., 2025), which shows that our proposed model can provide a stable and accurate generation for robot planning.

Table 8: **IoU score of robot mask.** We collect 5 key frames per trajectory and 40 camera views per frame, and evaluate the IoU between ground truth robot mask and generated robot mask.

	StackCube-v1 $\uparrow$	PullCubeTool-v1 $\uparrow$	PegInsertionSide-v1 $\uparrow$	Average $\uparrow$
Wan-2.1	0.7423	0.7189	0.7148	0.7253
TesserAct	0.8302	0.7704	0.7741	0.7915
OpenSora-1.3	0.7789	0.6956	0.6945	0.7229
Ours	<b>0.9091</b>	<b>0.9334</b>	<b>0.8970</b>	<b>0.9132</b>