

Visualization of Individually Sampled Data

David Stibbe

Thesis Doctoraal Computational Science

Supervisor: dr. Simon Portegies Zwart

Section Computational Science

and

Sterrenkundig Instituut "Anton Pannekoek"

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

Universiteit van Amsterdam

Kruislaan 403

1098 SJ

Amsterdam

July 13, 2006

Abstract

Till this day, star dynamics simulations are sampled using the snapshot sampling scheme, which works perfectly fine, but requires a lot of diskspace since for each snapshot the entire simulated system is being stored. In combination with current integration schemes, which are usually based on the individual time-step integration scheme, it also produces a computational overhead, because to be able to produce a snapshot the integrated system needs to be synchronized, i.e. all particles need to be interpolated to the same time instance. I propose an alternative sampling scheme called the individual time-step scheme. In this scheme particles are individually sampled, based on their individual integration steps. Using some basic, well known astrophysical models, I will compare these two sampling schemes, showing that the quality/size ratio of data resulting from the individual sampling scheme is better than that of data resulting from the snapshot sampling scheme.

Starshow is an application, built as a reference implementation for the visualization of the results of star dynamics simulations, capable of visualizing snapshot sampled data as well as individually sampled data. It was build specifically for the Personal Space Station, a semi-immersive, near-field, virtual environment, which allows for alternative methods of interaction with the visualized simulation.

Contents

1	Introduction	1
1.1	Star Dynamics Simulations and sampling	1
1.2	Visualization	1
1.3	This thesis	2
2	N-body Integration	3
2.1	Shared Time-step Integration	4
2.2	Individual Time-step Integration	4
3	Sampling Schemes	6
3.1	Snapshot Sampling Scheme	6
3.2	Individual Sampling Scheme	7
3.3	Measurement	7
4	Comparison between Snapshot and Individual Sampling	9
4.1	Positional Error	15
4.2	Quality	18
4.3	Energy Error	23
4.4	Required Disk Space	28
4.5	Required Disk Space vs Quality	32
5	Starshow	35
5.1	The Personal Space Station	35
5.1.1	History	36
5.1.2	Workings	37
5.2	The Idea behind Starshow	39
5.3	Interaction	39
5.4	Selecting Stars	41
5.4.1	Selection mechanism	41
5.4.2	Preliminary test	43
5.5	Visualizing Stars	45
5.5.1	Visualizing stars over time	45
5.5.2	Multi-threading	48
5.5.3	Buffering	49
5.6	Menus	49
5.6.1	Floating Menu	49
5.6.2	Palette Menu	49
5.6.3	Data window	51

5.7	Starshow's Internal Data Structure	51
5.8	Used Libraries	52
5.8.1	CAVElib	52
5.8.2	OpenGL	52
5.8.3	Starlab	53
5.8.4	STL	53
6	Discussion and Conclusion	57
6.1	Discussion	57
6.1.1	Sampling Schemes	57
6.1.2	Starshow	58
6.2	Conclusion	58
6.3	Future Work	59
6.3.1	Individual Sampling Scheme	59
6.3.2	Starshow	59

1 Chapter 1

2 Introduction

3 1.1 Star Dynamics Simulations and sampling

4 The branch of astrophysics describing the collective motions of stars subject to
5 their mutual gravity is called star dynamics. Simulations in star dynamics are
6 performed with the use of integrators. Over the course of time these integrators
7 have gradually evolved and quite a number of techniques have been developed
8 to optimize the integration methods [5]. Since storing the complete integration
9 usually requires too much space the results need to be sampled. However, these
10 sampling methods have hardly evolved and the most commonly used method
11 has always been and still is the snapshot sampling scheme. Although this sam-
12 pling scheme is not erroneous in any way, the size of the results obtained using
13 this method tend to be very large. In this thesis I will propose the individual
14 sampling method; a method that is, in combination with current integration
15 schemes, more efficient than the snapshot scheme in terms of the size of its
16 output and the amount of processing required to produce that output.

17 1.2 Visualization

18 Visualization helps scientists to gain a better understanding of the subject they
19 are examining. Although there are for astrophysicists already several visualizers
20 for star dynamics simulations available I developed a new application named
21 Starshow. This visualizer distinguishes itself from the other applications by
22 being designed specifically for the Personal Space Station [10], an experimental
23 virtual reality system, instead for the standard desktop computer. The Personal
24 Space Station is different than most virtually reality system because it is small
25 enough to fit on a desktop. Since a virtual reality system that can fit on a
26 desktop might become more common than large virtual reality systems such as
27 the CAVE [7], which requires an enormous amount of space, researchers might
28 be more inclined to use it. The controllers used for interaction with the PSS
29 are also quite novel, which allows for new ideas with regard to interaction with
30 the visualized simulations.

31 **1.3 This thesis**

32 This thesis consists of two parts: in the first part, chapter 2 till chapter 4, I
33 will examine the advantages of an individual sampling scheme over a snapshot
34 sampling scheme. In the second part, chapter 5, I will discuss the tool called
35 Starshow I've created for the PSS for the visualization of n-body simulations;
36 in particular those sampled with the individual sampling scheme with Starlab.
37 Finally, in chapter 6 I will discuss the findings I made during the course of this
38 thesis.

39 Chapter 2

40 N-body Integration

41 The n -body problem is the problem of determining the motions of n bodies,
42 given their initial positions, masses and velocities. Numerical integration in
43 time is used in physical simulation to integrate position, velocity, acceleration
44 and other state derivatives during body movement. N-body integration is the
45 numerical integration in time of a stellar system containing n number of particles
46 (bodies).

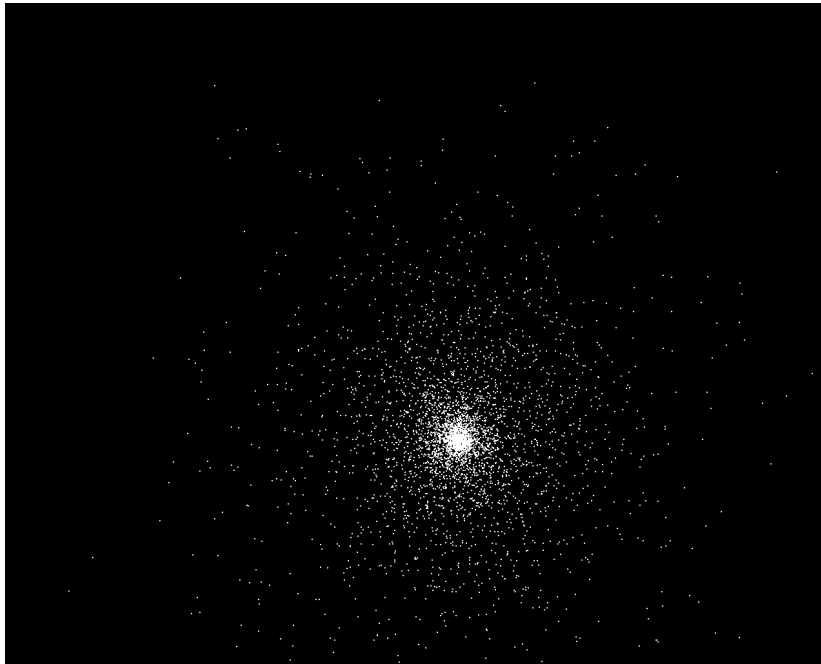


Figure 2.1: A star cluster.

47 One of the most important aspects of the n -body integrations is the integra-
48 tion time-step, the period of time in between two moments where the properties

49 of a star are redetermined. Since a larger time-step means less calculations, the
50 time-steps should be made as big as possible. However, the fact that a larger
51 time-step usually means a larger error makes finding an optimal integration
52 scheme a challenging task.

53 Over the years various methods have been developed to deal with the way
54 how the time-steps of the particles are determined. In this chapter I will discuss
55 two very well known integration schemes that use very distinct methods to
56 determine the time-step: the shared time-step integration and the individual
57 time-step integration.

58 2.1 Shared Time-step Integration

59 The shared time-step integration [3] [4] uses a very simple time-step scheme as
60 is shown in Illustration A in figure 2.2. In this scheme all particles have the
61 same constant time-step during the entire integration. The simplicity of this
62 scheme is its greatest advantage. However, it does not provide much flexibility
63 for the integrator and a high precision has a very high computation cost.

64 Shown in illustration B in figure 2.2 is a variation on the shared time-step
65 integration, called the variable time-step integration; all particles still have an
66 equal time-step. However, this time-step is now variable over time and rede-
67 termined at each integration step. This provides an increased flexibility while
68 maintaining most of the simplicity of the shared time-step integration.

69 2.2 Individual Time-step Integration

70 The individual time-step scheme is shown in illustration C in figure 2.2; for each
71 particle being integrated there is an individual time-step that is variable over
72 time. For each particle the size of this time-step will be redetermined at each of
73 the particle's integration steps. This scheme has the effect that the individual
74 time-step will be larger when a particle trajectory is stable and shorter when it
75 is less stable, which has the result that the overall precision of the integration
76 will be increased while the number of calculations is more or less maintained.

77 In illustration D in figure 2.2 a variation on the individual time-step scheme
78 called the block-timestep scheme [12] is shown; instead of allowing the individual
79 time-step to have any possible size, the time-step's size is restricted to be a
80 multiple of a certain minimum size. This way, the probability of having multiple
81 particles needing to be integrated simultaneously increases, resulting in a lower
82 computational cost.

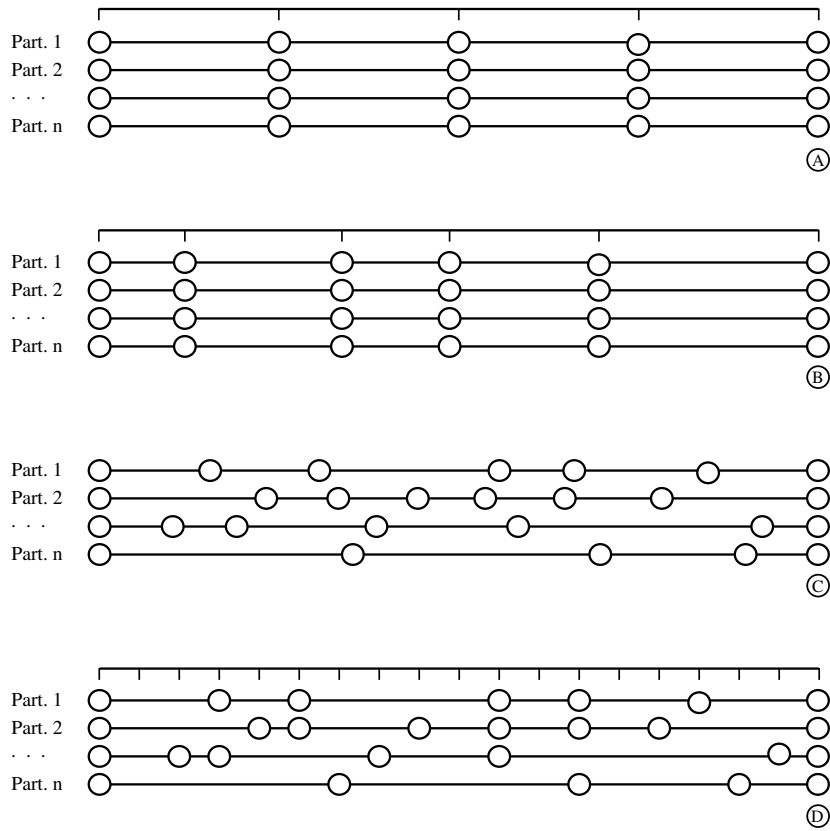


Figure 2.2: Illustration of the various integration methods: A) Shared time-step, B) Variable time-step, C) Individual time-step and D) blocked individual time-step

83 Chapter 3

84 Sampling Schemes

85 The user performing a simulation usually wants to store not only the end-state of
86 the simulation, but preferably the whole simulation from beginning to end. But
87 since storing the entire simulation would produce too much data, the simulation
88 needs to be sampled.

89 In this chapter I'll discuss the two main sampling schemes: the snapshot
90 sampling scheme and the individual sampling scheme. Both schemes will be
91 discussed using the shared and individual time-step integration schemes.

92 3.1 Snapshot Sampling Scheme

93 A snapshot is an instantiation of the entire system that is being modelled at
94 a certain time instance. When using the snapshot sampling scheme snapshots
95 are created at set time-intervals, which are independent from the integration
96 time-steps.

97 The snapshot sampling scheme is used to store the results of simulations since
98 the very beginning of star dynamics simulations. When using a shared time-
99 step integration scheme for the simulation, the use of snapshots makes perfect
100 sense, since at every integration time-step the data of the entire system being
101 modeled is already known; therefore at those time instances no additional action
102 to obtain a snapshot is needed. However, as graph A in figure 3.1 shows, often
103 the integration is sampled at time-intervals different from the time-steps of the
104 integrator. As a result the integrator still has to synchronize to the sampling
105 time-instances in order to generate a snapshot.

106 When sampling an individual time-step integration using a snapshot sampling
107 scheme, the integrator will always need to synchronize to the sampling time-
108 instances which can be seen in graph B in figure 3.1.

109 The snapshot sampling scheme has the advantage that it is very simple to
110 perform and when snapshot data is available, it can be re-used without much
111 effort. However, there are more disadvantages than advantages. The first disad-
112 vantage is that the resulting size is quite large, since it outputs the entire system
113 being modelled every time-interval. The second disadvantage is that it is very

114 difficult to determine beforehand the optimal time-interval; with the exception
115 of the basic shared time-step integration scheme, it is unknown at which sample
116 interval the precision of the integrator can be achieved. The third disadvantage
117 is that it almost always requires an extra synchronization step to generate a
118 snapshot.

119 3.2 Individual Sampling Scheme

120 Instead of taking whole snapshots at certain time intervals, one can sample each
121 particle that is being at integrated at each of its n^{th} integration steps. We refer
122 to this sampling scheme as the individual sampling scheme.

123 When applying this sampling scheme while using a shared or variable time-
124 step integration scheme the resulting data are snapshots, making the result look
125 somewhat similar to the snapshot sampling scheme as is shown in graph C in
126 figure 3.1.

127 Unlike with the snapshot sampling scheme the samples generated by this
128 scheme give a very good representation of the detail present in the integrator
129 as is shown in graph D in figure 3.1. It is possible to store the exact internal
130 representation of the integrator using this scheme.

131 The greatest disadvantage of the individual sampling scheme is that, when
132 used in combination with the individual integration scheme, the snapshot of a
133 system can not be read immediately from the output; some processing would
134 be required first.

135 The advantages however mostly outweigh this disadvantage: first, since the
136 individual sampling scheme does not need to synchronize all particles to a certain
137 time instance it requires no extra processing cost to generate the sampling data.
138 The second advantage is that this sampling scheme can always obtain a precision
139 equal to that of the integrator being used by sampling with a sample interval of
140 1. This also means that the sampling interval is an indication of the sampling
141 precision in relation to the integrator. The user can thus make an educated
142 guess of the precision of the results beforehand. The third advantage is that the
143 amount of sampled data is directly related to the number of integration steps
144 made by the integrator, which means that stable models require less disk space
145 than more unstable models. Last, but not least, is the fact that in contrast
146 with the snapshot sampling scheme, it is not possible to sample more than the
147 precision the used integrator provides, thereby preventing overhead in disk space
148 and calculations.

149 3.3 Measurement

150 Due to the different methodologies of the two sampling schemes, the precision
151 with which they sample are measured differently. The sample interval of the
152 snapshot sampling scheme is measured in time-units; in this thesis these are n-
153 body time-units, since output generated by the Starlab package is used, which
154 uses this as their time-unit. The sample interval of the individual sampling

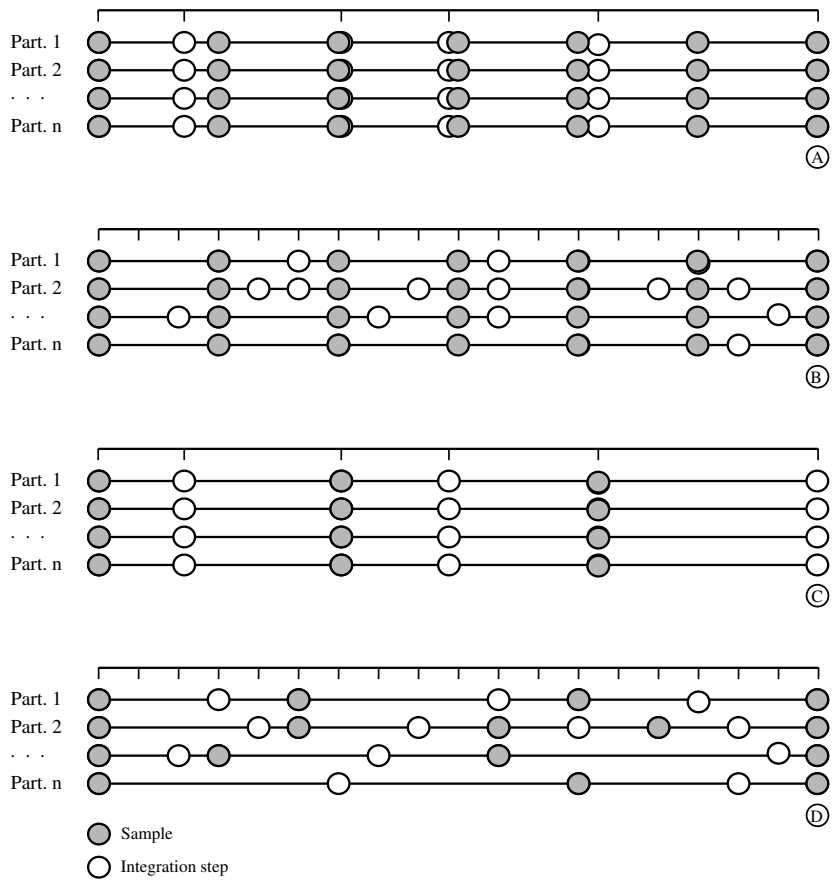


Figure 3.1: Illustration of the sampling methods: Snapshot sampling: A) Variable time-step integration, B) Individual time-step integration; Individual sampling: C) Variable time-step integration and D) individual time-step integration

155 scheme however is measured in number of integration steps; this means that a
 156 sample interval of n means that a sample is taken at every n integration steps.

157 Chapter 4

158 Comparison between 159 Snapshot and Individual 160 Sampling

161 In this chapter I will compare the snapshot and individual sampling schemes
162 when using an individual time-step integration scheme; this comparison will be
163 done based on their positional error, energy, size and overall quality.

164 The stellar dynamics data used in this thesis are all generated using the kira
165 integrator from the Starlab software package [14], which uses a block-timestep
166 integration scheme and supports output generated by snapshot sampling as well
167 as individual sampling. The stellar model being integrated is a King [11] model,
168 which is a distribution function often used by astrophysicists to model globular
169 clusters¹. King distributions are defined by 2 parameters: the number of stars
170 n and the non-dimensional central potential W_0 . The value of W_0 , ranging from
171 1 to 12, gives an indication on how uniform the stellar distribution is, where a
172 low W_0 leads to a more uniformly distributed system than a high W_0 . For the
173 following tests we will generally use King models with $n = 1000$ and $W_0 = 3$,
174 $W_0 = 6$ and $W_0 = 9$ integrated using kira over the period of 1 n-body time
175 unit² [9].

176 To indicate the precision of the data, we will refer to snapshot sampled data
177 with $s2^n$, where 2^n indicates the sample interval (in n-body time units) with
178 which the data was sampled. Individually sampled data will be referred to with
179 xn , where n indicates the sample interval.

180 The reason that the snapshot sample interval is indicated with a power of 2,
181 is because the kira integrator performs block-time individual integration with
182 timesteps that are a power of 2 and³ therefore requires the snapshot to be taken
183 at intervals with the power of 2 also.

184 The individual sampling scheme has the highest possible precision when
185 sampling at sample interval $x1$; at this interval the precision of the samples

¹ A globular cluster is a type of star cluster that has a spherical shape and contains between 10^4 to 10^6 stars and has an age of 10 to 16 billion years

² 1 n-body time unit $\approx 0.03110^6$ years

³ This is to ensure a more efficient use of the GRAPE [12]

186 is equal to the precision at which the integrator is integrating. Therefore, in
187 the coming comparisons, data obtained using individual sampling with sample-
188 interval $x1$, will be used as a base reference to compare the results of the other
189 sampled trajectories with; this data will be referred to as $x1$ data.

190 Figure 4.1 shows the trajectory of a particle with id 1 that is interacting with
191 2 other particles. These trajectories were created using a $W_0 = 12$ King model
192 consisting of 12 particles integrated over 10 n-body time units, with the shown
193 trajectories being a subset of these 12 particles for the time range $t \approx 2$ to $t \approx 3$
194 n-body time units. In figure 4.2 the same trajectories are shown, with their
195 x-position plotted against time. In the following sections the trajectory of the
196 particle with id 1 will be used for demonstration purposes.

197 Figures 4.3 and 4.4 show the effect the two sampling methods have on the
198 reconstruction of the trajectory of the particle with id 1. The trajectory is
199 reconstructed based on the samples using a 2^{nd} order hermite interpolation
200 (described in chapter 5.5.1). Figure 4.3 shows the effect of snapshot sampling
201 at $s2^{-2}$, $s2^{-1}$ and $s2^0$. Figure 4.4 shows the effect of individual sampling with
202 intervals of $x160$, $x320$ and $x640$ respectively.

203 There is a specific reason these sample intervals were chosen. Given the
204 example trajectory, the sample interval of $x160$ results in a trajectory that is
205 visibly distinct from the $x1$ trajectory while having about the same number
206 of samples as a snapshot sampled trajectory with sample interval $s2^2$. The
207 other sample intervals for snapshot sampling are $s2^1$ and $s2^0$ with corresponding
208 individual sampling intervals of $x320$ and $x640$ respectively.

209 The purpose of the figures 4.3 and 4.4 is to show that, even if both sam-
210 pling schemes use about the same amount of samples, the resulting interpolated
211 trajectories can be significantly different.

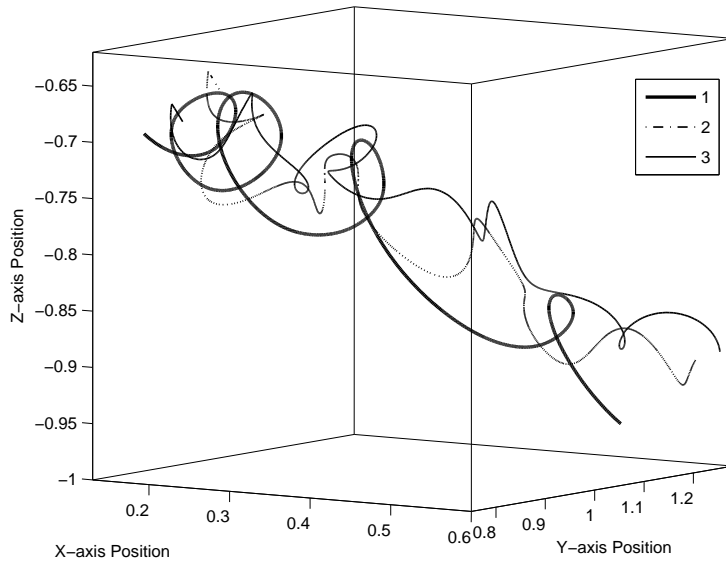


Figure 4.1: The trajectory of particle 1 together with the trajectories of two interacting particles. The trajectory of particle 1, indicated by the solid line, is used for demonstration purposes throughout this chapter. The data for the trajectories was generated by individual sampling with an interval of $x1$ of a $W_0 = 12$ King model consisting of 12 particles, integrated over 10 n-body time units. The trajectories are shown for time interval $t \approx 2$ till $t \approx 3$. Their lines were drawn by linear interpolation of the data points.

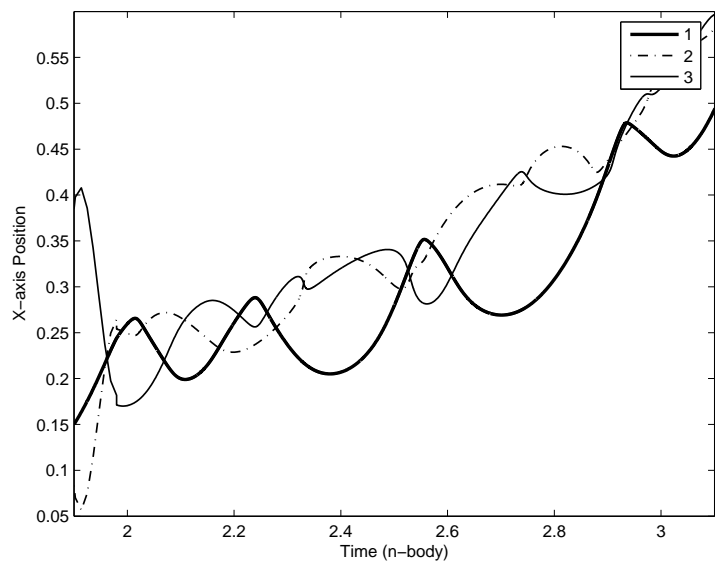


Figure 4.2: The x-axis positions of the trajectories shown in figure 4.1 plotted against time. The lines were drawn using linear interpolation.

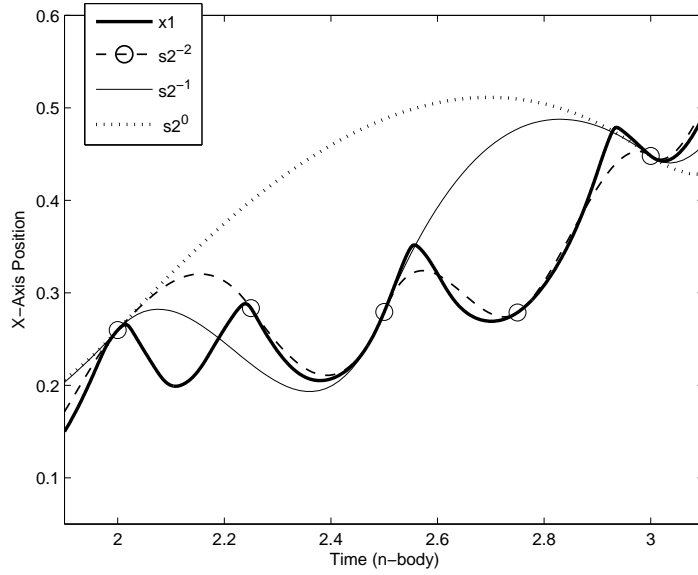


Figure 4.3: The effect of snapshot sampling the trajectory of particle 1, shown in figure 4.1. The thick solid line shows the interpolated trajectory based on data sampled with individual sampling interval x_1 as was shown in figure 4.2. The dotted line shows the interpolated trajectory based on data sampled with snapshot sample interval s_2^{-0} , the thin solid line for s_2^{-1} and the dashed line for s_2^{-2} . The markers for s_2^{-2} show the sample points that are obtained when sampling at that interval. The lines were drawn using a 2^{nd} order Hermite interpolation.

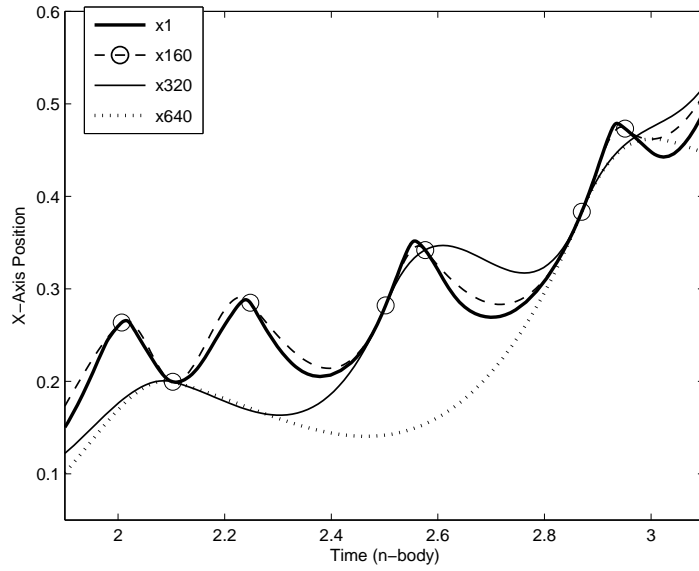


Figure 4.4: The effect of individually sampling the trajectory of particle 1, shown in figure 4.1. The thick solid line shows the interpolated trajectory based on data sampled with individual sampling interval $x1$ as was shown in figure 4.2. The dotted line shows the interpolated trajectory based on data sampled with individual sample interval $x640$, the thin solid line for $x320$ and the dashed line for $x160$. The markers for $x160$ show the sample points that are obtained when sampling at that interval. The lines were drawn using a 2^{nd} order Hermite interpolation.

212 4.1 Positional Error

213 With the term positional error $|\Delta d|$ we refer to the absolute difference in position
214 between the original trajectory and a reconstructed trajectory. This positional
215 error is determined as follows: for a given simulation, we obtain the x_1 data
216 and data sampled using either of the two sampling schemes. Then by using a
217 2^{nd} order Hermite interpolation we interpolate the sampled data to the time
218 instances for which x_1 data exists. The positional error is then the absolute
219 difference between the interpolated position and the corresponding x_1 position.

220 Figures 4.5 and 4.6 show the positional error $|\Delta d_x|$ over the x -axis of the
221 example trajectory for both sampling schemes for the same sample intervals as
222 were used in figures 4.3 and 4.4 respectively.

223 Figures 4.5 and 4.6 show various dips in their graphs; these dips occur when
224 the interpolated trajectory is equal to or intersects the x_1 trajectory, thus when
225 the positional error is minimal. It is noticeable however, that for individual
226 sampling almost all dips in the positional error have a minimum that is equal to
227 or lower than 10^{-6} , while for snapshot sampling this number is far less. This can
228 be explained by the fact that the sampling points generated for the individual
229 sampling are a subset of the x_1 samples, thus resulting at those points in a
230 positional error of 0. Samples generated using snapshot sampling on the other
231 hand hardly ever coincide with the x_1 samples.

232 Figure 4.6 clearly shows that in general, when the individual sampling interval
233 is divided by 2, the positional error decreases and the period between two
234 dips is divided in two new periods separated by a dip, i.e. the extra sample.
235 However, this does ignore the ‘extra’ dips caused by the interpolated trajectory
236 intersecting the x_1 trajectory.

237 Figure 4.5 shows, just as figure 4.6 for individual sampling, that when the
238 sample interval is divided by 2, the period between two dips is divided in two
239 new periods divided by a dip. Again, this is when the ‘extra’ dips caused by
240 the interpolated trajectory intersecting the x_1 trajectory are ignored.

241 The difference between figure 4.5 and 4.6 however is that the dips in the
242 graph of the snapshot sampled data occur at regular time intervals (for data
243 with sample interval s_2^{-2} every 0.25 n-body time units for example), while the
244 time interval between dips for individual sampled data is completely irregular.

245 Both figures show a maximum positional error of roughly $|\Delta d_x| \approx 0.2$, how-
246 ever, this is a mere happenstance. The positional error can easily exceed 1.

247 The quality of the positional error depends on the velocity of a particle; a
248 particle that has a high velocity with a certain positional error can be considered
249 ‘better’ than a particle that has the same positional error, but a lower velocity.
250 Therefore, even though the graphs of the positional error can give a person a
251 good insight in the effects of the various sampling intervals on the deviation
252 of the position, it is not a good indication of the quality of the interpolated
253 trajectories. The positional error of a trajectory alone is simply not enough as
254 a quality measurement since the quality of the positional error depends on the
255 velocity of the particle.

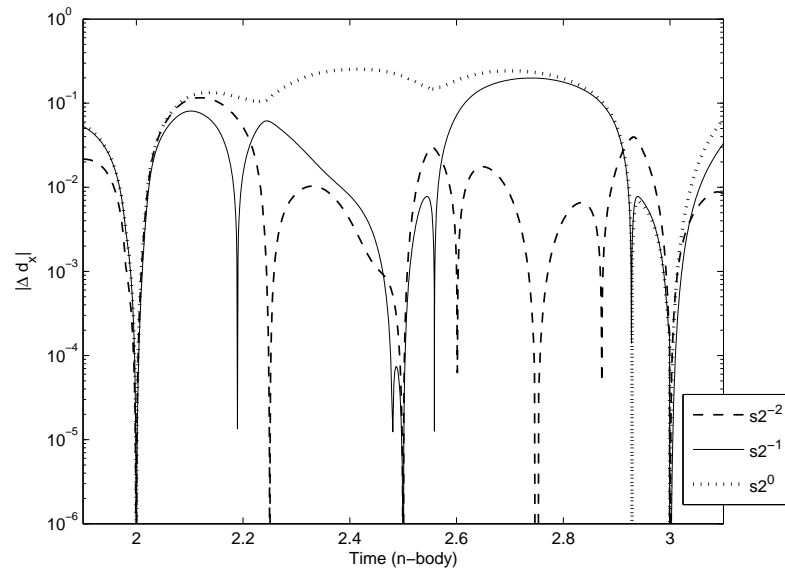


Figure 4.5: Positional error $|\Delta d_x|$ over the x-axis for the snapshot sampled trajectories shown in figure 4.3 with respect to the x_1 trajectory. The lines are linearly interpolated.

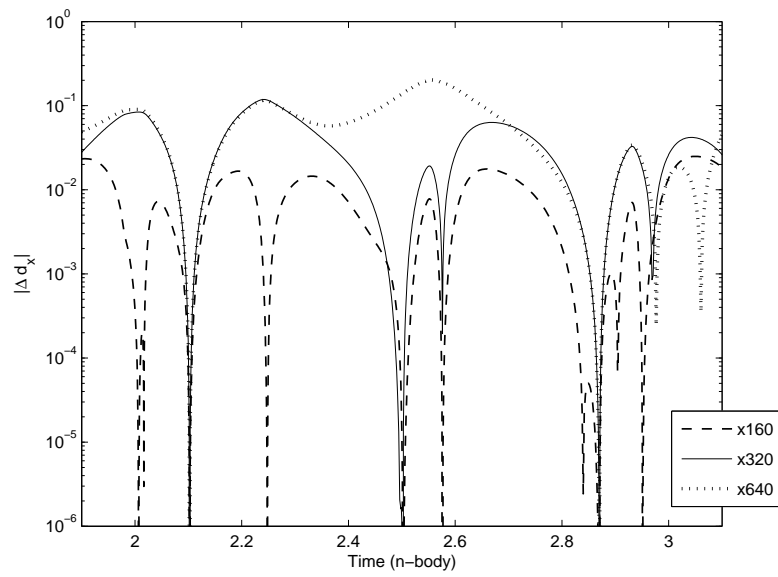


Figure 4.6: Positional error $|\Delta d_x|$ over the x-axis for the individually sampled trajectories shown in figure 4.4 with respect to the x_1 trajectory. The lines are linearly interpolated.

256 4.2 Quality

257 As it was said before: the positional error is insufficient for the comparison
 258 between trajectories. Therefore a better measurement to determine the quality
 259 of a trajectory is needed. To make comparison of results easier, this quality value
 260 (Q) should be in the range $[0, 1]$ with 1 being a perfect trajectory. The method
 261 decided upon was to use the positional error ($\|\Delta d\|$) scaled by the velocity of
 262 the x_1 trajectory ($\|v_{x_1}\|$):

$$Q = \frac{1}{\|v_{x_1}\| \log \|\Delta d\|} \quad (4.1)$$

263 Equation 4.1 however has a few drawbacks. The first issue is that the value
 264 of Q can become negative when $0 < \|v_{x_1}\| < 1$ or $0 < \|\Delta d\| < 1$. Therefore,
 265 we increase both $\|v_{x_1}\|$ and $\|\Delta d\|$ with 1, resulting in:

$$Q = \frac{1}{\|v_{x_1}+1\| \log (\|\Delta d\| + 1)} \quad (4.2)$$

however, this does not solve the second issue:

$$\lim_{\Delta d \rightarrow 0} Q = \lim_{\Delta d \rightarrow 0} \frac{1}{\|v_{x_1}+1\| \log (\|\Delta d\| + 1)} = \frac{1}{0} = \infty \quad (4.3)$$

266 To solve this problem, we increase $\|v_{x_1}+1\| \log (\|\Delta d\| + 1)$ with 1 resulting in:

$$\begin{aligned} Q &= \frac{1}{\|v_{x_1}+1\| \log (\|\Delta d\| + 1) + 1} \\ &= \frac{\log (\|v_{x_1}\| + 1)}{\log (\|\Delta d\| + 1) + \log (\|v_{x_1}\| + 1)} \end{aligned} \quad (4.4)$$

267 This value of Q ranges between $[0, 1]$ for all $\|\Delta d\|$ and $\|v_{x_1}\|$; however, when
 $\|v_{x_1}\| = 0$, then Q will always be 0, even when $\|\Delta d\| = 0$. Therefore, when
 determining Q the following rule needs to be followed:

$$\text{when } \|\Delta d\| = 0 \quad \text{then } Q = 1 \quad (4.5)$$

268 Figures 4.7 and 4.8 illustrate the effect of this quality quantifier Q over the x -
 269 axis on the trajectories from figures 4.3 and 4.4 respectively, showing a noticeable
 270 difference in the distributions of the Q value between the two sampling schemes.
 271 This is very different from what we saw for the positional error in figures 4.5
 272 and 4.6 where the graphs were far less distinct.

273 The quality graphs are 1 at those time-instances where either the sampled
 274 data set has a sample or where the interpolated trajectory intersects the x_1
 275 trajectory; very much like the ‘dips’ in the graphs of the positional error.

276 Figure 4.9 illustrates the log plot of $1 - Q$ versus the sampling interval of the
 277 interpolated snapshot sampled data, while table 4.1 shows the actual data. The
 278 models used were $W_0 = 3$, $W_0 = 6$ and $W_0 = 9$ King models containing 1000
 279 particles integrated over 1 n-body time unit; the quality was averaged over 300
 280 tests per model. The bars indicate the standard deviation of Q .

281 The figure clearly shows that the log plots of all 3 models are linear in log
 282 space with more or less the same slopes. A least-squares fit in log space for

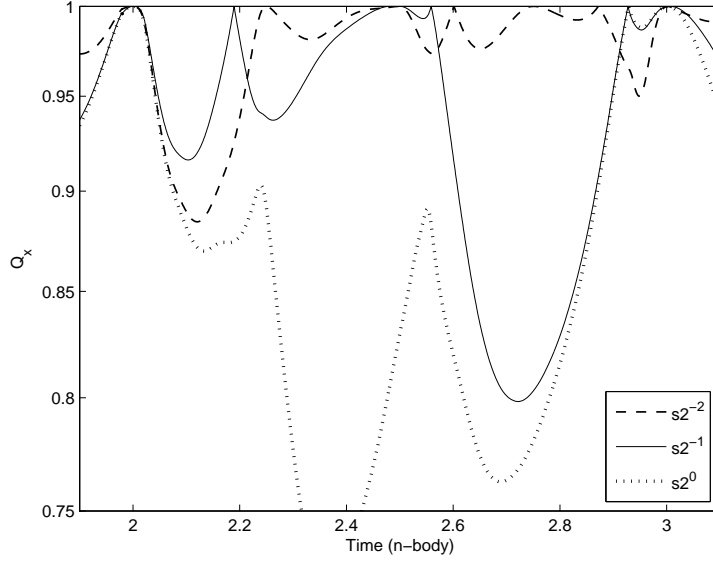


Figure 4.7: Quality Q_x over the x-axis for the snapshot sampled trajectories shown in figure 4.3. The lines are linearly interpolated.

283 each model results in the equations 4.6, 4.7 and 4.8, where the variable n is
 284 the snapshot sampling interval. From these fits in log space we can derive the
 285 equations 4.9, 4.10 and 4.11 respectively.

286 This means that, for example, when the snapshot sample interval of a $W_0 = 9$
 287 King model decreases 2 times, the value of $1 - Q$ decreases $2^{1.71}$. This can be
 288 easily verified with table 4.1 when you consider that $2^{1.77} \approx 10^{0.53}$, $2^{1.75} \approx 10^{0.53}$
 289 and $2^{1.71} \approx 10^{0.51}$.

	$s2^0$	$s2^{-1}$	$s2^{-2}$	$s2^{-3}$	$s2^{-4}$
$W_0 = 3$	$10^{-2.07}$	$10^{-2.59}$	$10^{-3.09}$	$10^{-3.60}$	$10^{-4.14}$
$W_0 = 6$	$10^{-1.92}$	$10^{-2.49}$	$10^{-2.99}$	$10^{-3.49}$	$10^{-4.01}$
$W_0 = 9$	$10^{-1.48}$	$10^{-2.01}$	$10^{-2.62}$	$10^{-3.16}$	$10^{-3.65}$
	$s2^{-5}$	$s2^{-6}$	$s2^{-7}$	$s2^{-8}$	
$W_0 = 3$	$10^{-4.70}$	$10^{-5.24}$	$10^{-5.78}$	$10^{-6.30}$	
$W_0 = 6$	$10^{-4.55}$	$10^{-5.10}$	$10^{-5.63}$	$10^{-6.15}$	
$W_0 = 9$	$10^{-4.13}$	$10^{-4.62}$	$10^{-5.12}$	$10^{-5.62}$	

Table 4.1: Values of $1 - Q$ for snapshot sampling corresponding to figure 4.9

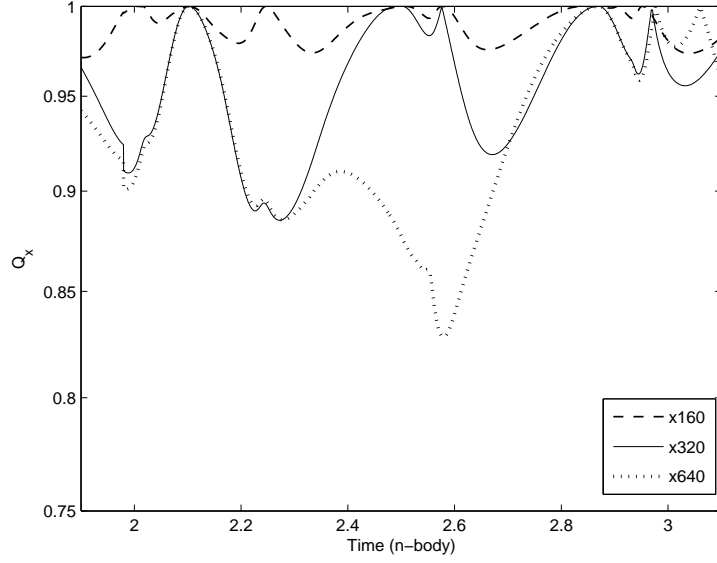


Figure 4.8: Quality Q_x over the x-axis for the individually sampled trajectories shown in figure 4.4. The lines are linearly interpolated.

$$\ln(1 - Q_{w3}^{snap}) \approx 1.77 \ln n - 4.70 \quad (4.6)$$

$$\ln(1 - Q_{w6}^{snap}) \approx 1.75 \ln n - 4.44 \quad (4.7)$$

$$\ln(1 - Q_{w9}^{snap}) \approx 1.71 \ln n - 3.55 \quad (4.8)$$

290 Figure 4.10 illustrates the log plot of $1 - Q$ versus the sampling interval of the
 291 interpolated individually sampled data, while table 4.2 shows the actual data.
 292 The models used were the same as for figure 4.9. Since the graphs in figure 4.10
 293 are not as linear in log space as the graphs in figure 4.9, we fit them to second
 294 order polynomials instead of linear functions.

295 The least-squares fit in log space for each model results in the equations 4.12,
 296 4.13 and 4.14, where the variable n is the individual sampling interval. From
 297 these fits in log space we can derive the equations 4.15, 4.16 and 4.17 respectively.

$$1 - Q_{w3}^{snap} \approx n^{1.77} e^{-4.70} \quad (4.9)$$

$$1 - Q_{w6}^{snap} \approx n^{1.75} e^{-4.44} \quad (4.10)$$

$$1 - Q_{w9}^{snap} \approx n^{1.71} e^{-3.55} \quad (4.11)$$

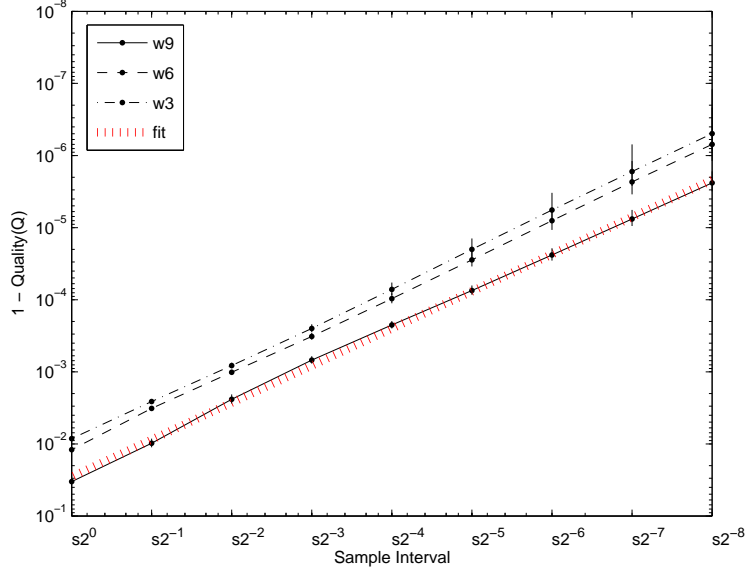


Figure 4.9: The $1 - Q$ value of interpolated trajectories based on snapshot sampled data, for various sample intervals ranging from $s2^0$ till $s2^{-8}$. The value of Q is the mean over 300 tests. The data sampled are King models consisting of 1000 particles with $W_0 = 3$, $W_0 = 6$ and $W_0 = 9$, integrated over 1 n-body time unit. The solid line indicates that the data sampled is a King model with $W_0 = 9$, the dashed line a King model with $W_0 = 6$ and the dash-dotted line a King model with $W_0 = 3$. Bullets indicate the actual measured values. The large dotted line is a least-square fitted function to $W_0 = 9$. The lines of the graphs are linearly interpolated. The vertical lines indicates the standard deviation of $1 - Q$.

	$x2$	$x4$	$x8$	$x16$	$x32$
$W_0 = 3$	$10^{-7.08}$	$10^{-6.20}$	$10^{-5.43}$	$10^{-4.56}$	$10^{-3.78}$
$W_0 = 6$	$10^{-7.04}$	$10^{-6.33}$	$10^{-5.50}$	$10^{-4.61}$	$10^{-3.81}$
$W_0 = 9$	$10^{-7.10}$	$10^{-6.38}$	$10^{-5.55}$	$10^{-4.67}$	$10^{-3.87}$
	$x64$	$x128$	$x160$	$x320$	$x640$
$W_0 = 3$	$10^{-3.16}$	$10^{-2.68}$	$10^{-2.55}$	$10^{-2.23}$	$10^{-2.08}$
$W_0 = 6$	$10^{-3.20}$	$10^{-2.71}$	$10^{-2.58}$	$10^{-2.21}$	$10^{-1.96}$
$W_0 = 9$	$10^{-3.26}$	$10^{-2.75}$	$10^{-2.61}$	$10^{-2.18}$	$10^{-1.79}$

Table 4.2: Values of $1 - Q$ for individual sampling corresponding to figure 4.10

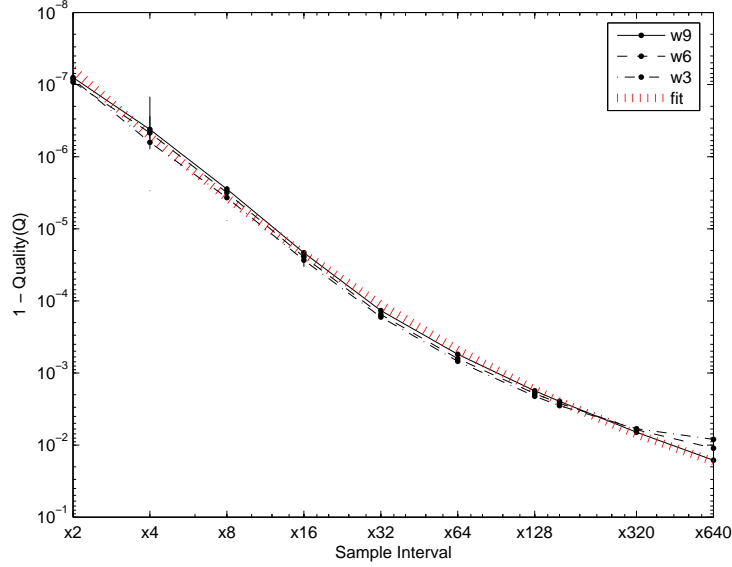


Figure 4.10: The $1 - Q$ value of interpolated trajectories based on individually sampled data, for various sample intervals ranging from $x2$ till $x640$. The value of Q is the mean over 300 tests. The data sampled are King models consisting of 1000 particles with $W_0 = 3$, $W_0 = 6$ and $W_0 = 9$, integrated over 1 n-body time unit. The solid line indicates that the data sampled is a King model with $W_0 = 9$, the dashed line a King model with $W_0 = 6$ and the dash-dotted line a King model with $W_0 = 3$. The large dotted line is a least-square fitted function to $W_0 = 9$. Bullets indicate the actual measured values. The lines of the graphs are linearly interpolated. The vertical lines indicates the standard deviation of $1 - Q$.

$$\ln(1 - Q_{w3}^{ind}) \approx -0.24 \ln n^2 + 3.81 \ln n - 19.04 \quad (4.12)$$

$$\ln(1 - Q_{w6}^{ind}) \approx -0.22 \ln n^2 + 3.63 \ln n - 18.96 \quad (4.13)$$

$$\ln(1 - Q_{w9}^{ind}) \approx -0.18 \ln n^2 + 3.46 \ln n - 18.92 \quad (4.14)$$

$$1 - Q_{w3}^{ind} \approx n^{-0.48+3.81} e^{-19.04} = n^{3.32} e^{-19.04} \quad (4.15)$$

$$1 - Q_{w6}^{ind} \approx n^{-0.44+3.63} e^{-18.96} = n^{3.21} e^{-18.96} \quad (4.16)$$

$$1 - Q_{w9}^{ind} \approx n^{-0.36+3.46} e^{-18.92} = n^{3.10} e^{-18.92} \quad (4.17)$$

298 4.3 Energy Error

299 Astrophysicists often want to know the energy of a system at a given time
300 instance, which needs to be constant throughout the simulation. Therefore it is
301 useful to determine how well the energy conservation of an interpolation based
302 on individual sampling actually is.

303 To be able to determine the energy at a given time for the entire system
304 that is being simulated, data is needed on all particles for that given time.
305 The energy error for individually sampled data is determined as follows: we
306 use snapshot sampled data with a very short sample interval as reference; for
307 each snapshot in the reference data, the individually sampled data needs to be
308 interpolated to the corresponding time instance. The energy error is the absolute
309 difference between the energy of the interpolated data and the corresponding
310 snapshot. To generate snapshots from an individually sampled data set, we use
311 the ‘worldline’ tool from the Starlab software package, while determining the
312 energy of a snapshot is done using the ‘energy’ tool from the Starlab package.

313 The models used were again 1000 particle King models with $W_0 = 9$, $W_0 = 6$,
314 and $W_0 = 3$. The energy and energy error are averaged over 100 runs for
315 various sample rates. The reference snapshot sample data was generated with
316 an interval of $s2^{-8}$.

317 Figure 4.11 shows the energy of the example trajectories that were shown
318 in figure 4.4 and the energy of the reference snapshots indicated as ‘base’. It is
319 interesting to note that the energy of the base snapshots also seems to deviate
320 at two occurrences. Figure 4.11 shows the absolute energy error of the same
321 example trajectories. Interesting to see is the fact that in this figure the energy
322 of the $x1$ data seems to deviate from that of the base, which is odd⁴, since the
323 $x1$ data should be equal to the data the integrator had when it generated the
324 reference snapshots.

325 Figure 4.13 shows for various sample intervals the mean energy of individually
326 sampled data for sample intervals ranging from $x1$ till $x640$. The data sampled
327 were King models containing 1000 particles with $W_0 = 3$, $W_0 = 6$ and $W_0 = 9$,
328 integrated over 1 n-body time units. The mean was taken over 100 runs. It
329 shows that for all models the energy remains around the -0.25 for the sample
330 intervals $\leq x64$. When the sample interval becomes larger than $x64$ the energy
331 starts to deviate noticeably.

332 Figure 4.14 shows the mean absolute error of the energy values shown in
333 figure 4.13. We fit the graphs of each model in figure 4.14 to a least-squares
334 linear function in log space resulting in the equations 4.18, 4.19 and 4.20, where
335 the variable n is the snapshots sampling interval. These fits in log space can be
336 deduced to equations 4.21, 4.22 and 4.23 respectively.

⁴perhaps this is caused by the the ‘worldline’ or ‘energy’ tool

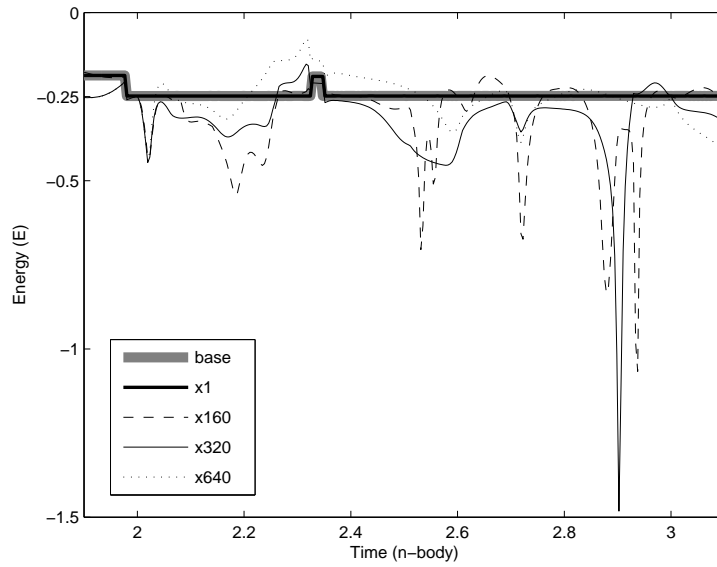


Figure 4.11: Energy of snapshots interpolated from the same individually sampled trajectories shown in figure 4.4. The trajectories were interpolated to snapshots at the same time instances as the ‘base’. The ‘base’ is shown as the gray line, indicating the energy of the snapshots generated when sampling the original trajectory at snapshot sampling interval 2^{-8} . The lines are linearly interpolated.

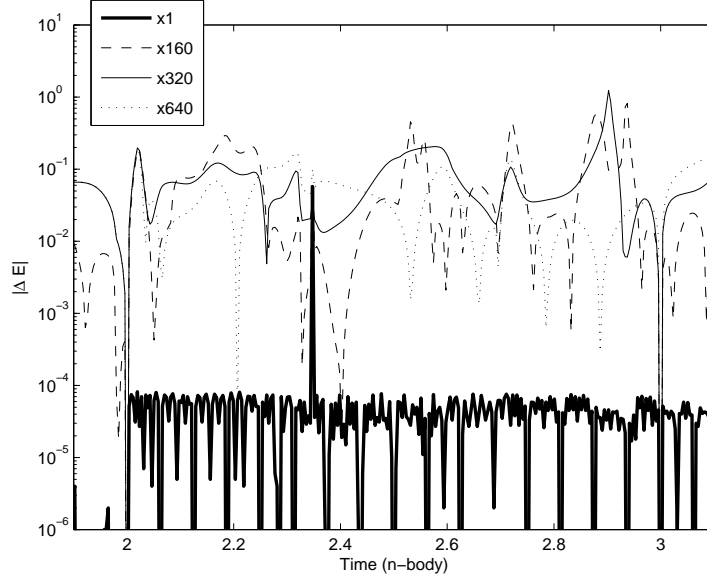


Figure 4.12: Absolute error of the trajectory energies shown in figure 4.11 with the respect to the ‘base’.The lines are linearly interpolated.

$$\ln |\Delta E|_{w3} \approx 2.29 \ln n - 19.80 \quad (4.18)$$

$$\ln |\Delta E|_{w6} \approx 2.02 \ln n - 18.32 \quad (4.19)$$

$$\ln |\Delta E|_{w9} \approx 2.21 \ln n - 18.97 \quad (4.20)$$

$$|\Delta E|_{w6} \approx n^{2.29} e^{-19.80} \quad (4.21)$$

$$|\Delta E|_{w6} \approx n^{2.02} e^{-18.32} \quad (4.22)$$

$$|\Delta E|_{w6} \approx n^{2.21} e^{-18.97} \quad (4.23)$$

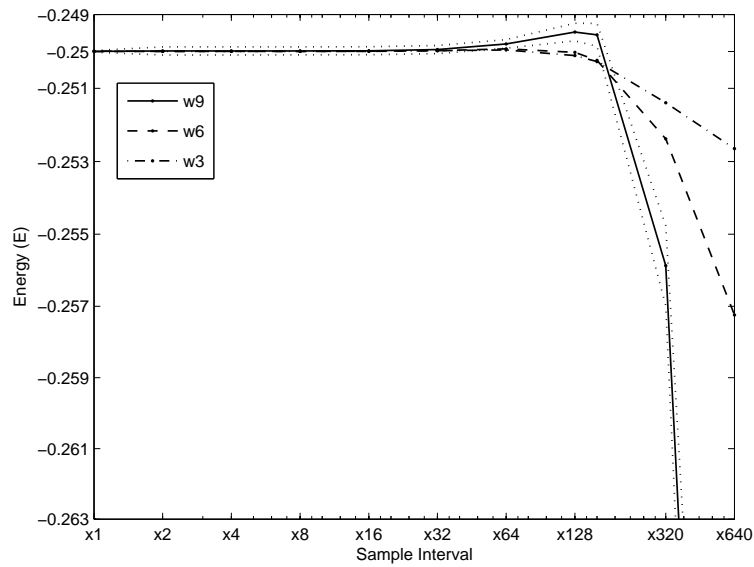


Figure 4.13: Mean energy of individually sampled data for sample intervals ranging from x_1 till x_{640} . The data sampled were King models containing 1000 particles with $W_0 = 3$, $W_0 = 6$ and $W_0 = 9$, integrated over 1 n-body time units. The mean was taken over 100 runs. The solid line indicates the $W_0 = 9$, the dashed line $W_0 = 6$ and the dash-dotted line $W_0 = 3$. The dotted lines above and below $W_0 = 9$ are indications of the standard deviation in the energy of $W_0 = 9$.

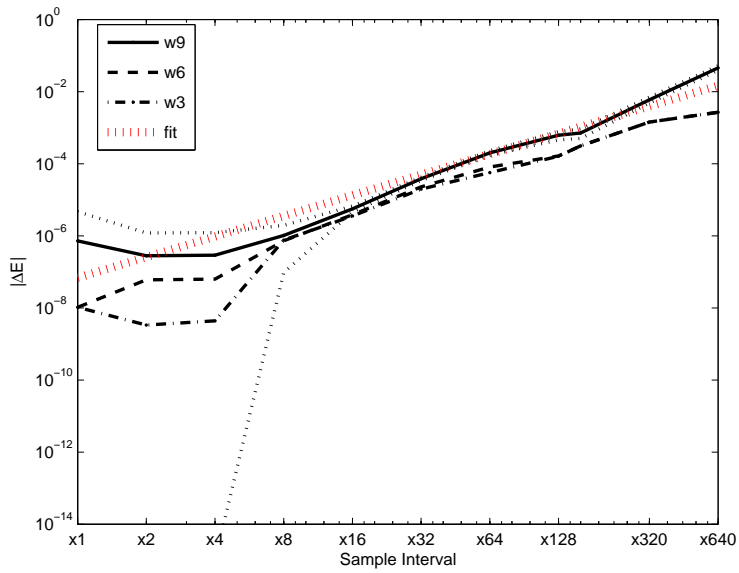


Figure 4.14: The mean absolute error of the energy values shown in figure 4.13. The dotted lines above and below $W_0 = 9$ are indications of the standard deviation in the energy error of data sampled from $W_0 = 9$. The large dotted line is a least-square fitted function to $W_0 = 9$.

337 4.4 Required Disk Space

338 When performing n-body simulations, the size of the resulting data can become
 339 incredibly large; therefore it is worth examining whether changing from using
 340 snapshot sampling to individual sampling results in a smaller size of the resulting
 341 data set.

342 We will determine the size of the sampled data set for the integration of
 343 King models with $W_0 = 3$, $W_0 = 6$ and $W_0 = 9$ consisting of 1000 particles,
 344 integrated over 1 n-body time unit. This is done with both sampling schemes
 345 for a range of sample intervals; for each sample interval the resulting size is the
 346 mean taken over 300 tests.

When performing snapshot sampling with kira, given a sample interval 2^m , with m being an integer, the number of snapshots the integrator generates, when integrating over t n-body time-units, is

$$n_s = t2^{-m} + 1 \quad (4.24)$$

347 however, when the sample interval $2^m \geq 1$, the integrator generates instead of
 348 $t2^{-m} + 1$ snapshots just $t2^{-m}$ snapshots⁵, because kira then doesn't generate any
 349 snapshot at $t = 0$. In the following graphs I compensated for this inconsistency
 350 by multiplying the resulting size of the data generated by interval $s2^0$ times 2
 351 ⁶.

The expected size of a data set resulting from the snapshot sampling scheme would be

$$s_{snap}(m, t) = s_s n_s = s_s(t2^{-m} + 1) \quad (4.25)$$

352 with s_s being the size of a single snapshot. For the integrated models discussed
 353 in this chapter this would result in $s_{snap}(m) = s_s(2^{-m} + 1)$.

354 Figure 4.15 and table 4.3 show that the average sizes of the snapshot sam-
 355 pling scheme for all 3 King models result in function 4.25. The size of a single
 356 snapshot for a system consisting of 1000 particles is $s_s \approx 0.5MB$.

	$s2^0$	$s2^{-1}$	$s2^{-2}$	$s2^{-3}$	$s2^{-4}$
$W_0 = 3$	1.1328	1.7000	2.8500	5.1492	9.7461
$W_0 = 6$	1.1328	1.7000	2.8484	5.1461	9.7414
$W_0 = 9$	1.1328	1.6992	2.8438	5.1398	9.7266
	$s2^{-5}$	$s2^{-6}$	$s2^{-7}$	$s2^{-8}$	
$W_0 = 3$	18.9398	37.3258	74.0977	147.6469	
$W_0 = 6$	18.9289	37.3055	74.0609	147.5687	
$W_0 = 9$	18.9008	37.2539	73.9516	147.3500	

Table 4.3: Sizes(MB) of the data sets resulting from snapshot sampling corresponding to figure 4.15

⁵ however, when $t \bmod 2^m \neq 0$ the number of snapshots is $t2^m + 1$

⁶ this is possible because the model is only integrating over 1 n-body time-unit for these tests

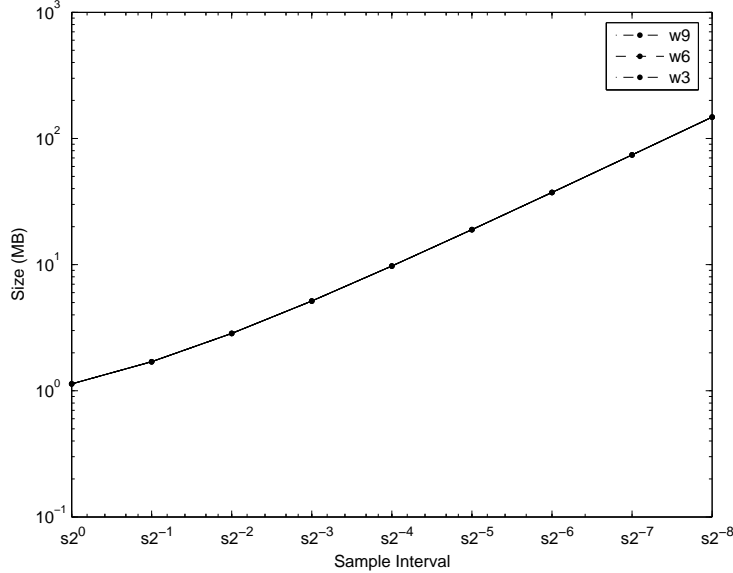


Figure 4.15: Average size of snapshot sampled trajectories for various sample intervals ranging from $s2^0$ till $s2^{-8}$ over 300 tests. The models sampled were King models containing 1000 particles, with $W_0 = 3$, $W_0 = 6$ and $W_0 = 9$, integrated over 1 n-body time unit. Bullets indicate the average sizes. The lines are linearly interpolated.

Since the required disk space depends on the number of samples, we expect the function of the average required disk space of individual sampled data sets to behave as:

$$s_{ind}(n) = \frac{s_{x1}}{n} \quad (4.26)$$

357 where s_{x1} is the disk space required for the data set generated at sampling
 358 interval $x1$ and n the individual sample interval.

359 Figure 4.16 and table 4.4 show the average required disk space of the indi-
 360 vidual sampling scheme. The graph and table show that the function 4.26 is
 361 correct until $n \approx 64$.

362 By default, kira produces a snapshot each n-body time unit when performing
 363 individual sampling. Thus when performing individual sampling, when a model
 364 is integrated over 1 n-body time unit, kira will produce 2 snapshots at least.

365 Kira produces less data per particle when performing individual sampling
 366 than when performing individual sampling, therefore the size of a snapshot made
 367 by kira when performing individual sampling differs from the size of a snapshots
 368 when performing snapshot sampling. A simple test with kira shows us that the
 369 size of two snapshots of 1000 particles when performing individual sampling is
 370 $2s_s \approx 0.5MB$.

371 When performing individual sampling with $n > 64$ the sampling interval
 372 starts to exceed the number of integration steps in the period of 1 body time-

373 unit of more and more stars, thus leaving those stars with only the samples at
 374 the snapshots. Therefore, the required disk space will approach the size of two
 375 snapshots.

376 In contrast with the snapshot sampling scheme, the difference between sizes
 377 of the different models is quite significant. For sample interval $x1$ the size of the
 378 $W_0 = 9$ model is about 1.5 times as big as the other two models. This is mostly
 379 caused by the fact that the stars in this model have far more interactions than
 380 the other two models, causing the integrator to take more integration steps for
 381 the calculation of this model and therefore results in a bigger sampled data set.

382 The figures also show that all scales between the sizes of the different models
 383 decrease as the size of the sample interval increases, which makes sense, since
 384 all sizes eventually converge towards the same $2s_s$, at which moment the scales
 385 between the sizes of the different models is thus 1.

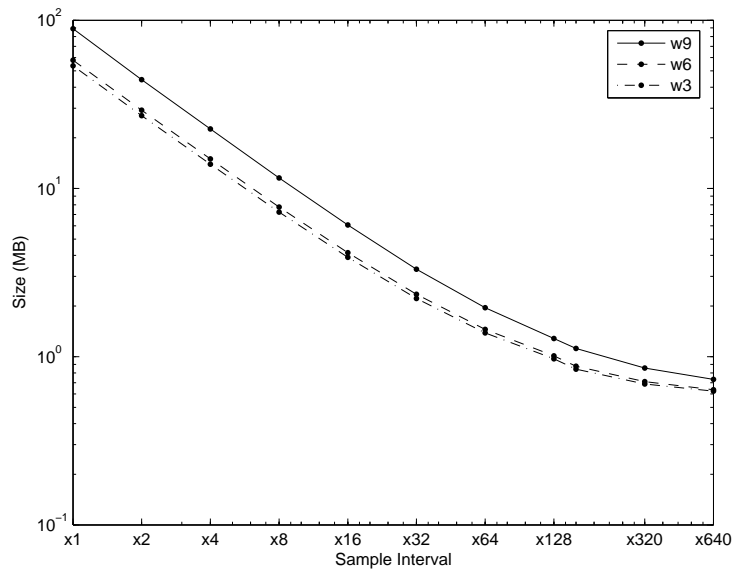


Figure 4.16: Average size of individually sampled trajectories for various sample intervals ranging from $x1$ till $x640$ over 300 tests. The models sampled were King models containing 1000 particles, with $W_0 = 3$, $W_0 = 6$ and $W_0 = 9$, integrated over 1 n-body time unit. The lines are linearly interpolated. The sizes resulting from the $W_0 = 9$ King model are indicated by the solid line, the $W_0 = 6$ model by the dashed line and the $W_0 = 3$ model by the dash-dotted line. Bullets indicate the average sizes.

	$x1$	$x2$	$x4$	$x8$	$x16$	
$W_0 = 3$	53.5976	27.0891	13.9343	7.2394	3.8935	
$W_0 = 6$	57.8242	29.1702	14.9755	7.7581	4.1506	
$W_0 = 9$	89.0738	44.3732	22.5845	11.5414	6.0679	
	$x32$	$x64$	$x128$	$x160$	$x320$	$x640$
$W_0 = 3$	2.2193	1.3845	0.9716	0.8437	0.6884	0.6239
$W_0 = 6$	2.3494	1.4526	1.0105	0.8783	0.7112	0.6354
$W_0 = 9$	3.3099	1.9566	1.2828	1.1187	0.8565	0.7326

Table 4.4: Size(MB) of the data sets resulting from individual sampling corresponding to figure 4.16

386 4.5 Required Disk Space vs Quality

387 In this section we will illustrate how the snapshot sampling compares to indi-
388 vidual sampling based on the size/quality ratio of their resulting data sets. In
389 figure 4.17, for both sampling schemes, the size of the resulting data sets as
390 determined in section 4.4 is plotted against their quality as was determined in
391 section 4.2. The figure shows a good comparison between the two sampling
392 schemes.

393 A least-squares fit to a linear function in log space for each snapshot sampled
394 model results in the equations 4.27, 4.28 and 4.29. From these fits in log space
395 we can derive the equations 4.30, 4.31 and 4.32 respectively.

396 It is clear that the slopes of all 3 graphs in log space are more or less equal;
397 the slope of all 3 graphs are around $-0,51$, which means that, when the value
398 of $1 - Q$ becomes 10 times larger (thus worse), the size of the resulting data set
399 is scaled by $10^{-0.51} \approx 0.31$.

400 A least-squares fit to a linear function in log space for each individually sam-
401 pled model results in the equations 4.33, 4.34 and 4.35. From these fits in log
402 space we can derive the equations 4.36, 4.37 and 4.38 respectively.

403 The slopes of all 3 graphs in log space are more or less equal to $-0,33$. This
404 means that, when the value of $1 - Q$ becomes 10 times larger, the size of the
405 resulting data set is scaled by $10^{-0.33} \approx 0.47$.

406 Both individual sampling and snapshot sampling have a relatively constant
407 slope, regardless of the type of model being sampled; thus the relation between
408 quality and size is hardly affected by the model.

409 The fitted functions indicate that if the quality is taken low enough, the
410 required disk space for snapshot sampled data will become lower than that of
411 the individually sampled data. However, these fitted functions do not take into
412 account the properties that were described in section 4.4, which is that, for
413 a large enough sample interval the required disk space for sampling using kira
414 will approach the size of two snapshots. However, when using kira, the size of a
415 snapshot consisting of a 1000 particles created using snapshot sampling is about
416 $0.5MB$, while a snapshot in a individually sampled data set is about $0.25MB$.
417 Looking at figure 4.17 you can see that the graphs for the disk space required
418 for individual sampling get below the $1MB$ (the size of two snapshots generated
419 using snapshot sampling) without crossing the graphs for snapshot sampling;
420 thus, at a given quality, the required disk space for individual sampling will
421 always be less than that required for snapshot sampling.

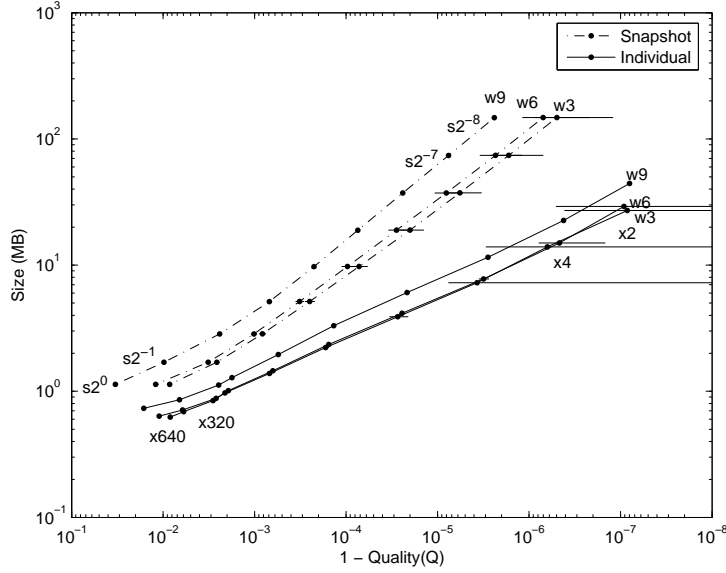


Figure 4.17: The size plotted against quality. The sizes are as were given in figures 4.15 and 4.16. The qualities are as were given in figures 4.9 and 4.10. The bullets indicate the sampling intervals, for individual sampling as well as snapshot sampling. Each line is linearly interpolated and represents a different sampling scheme/King model combination. Dashed lines represent snapshot sampled data and solid lines represent individually sampled data. Each sampling scheme has a separate line for each different King model: $W_0 = 9$, $W_0 = 6$ and $W_0 = 3$.

$$\ln(s_{w3}^{ind}) \approx -0.51 \ln(1 - Q_{w3}) - 2.48 \quad (4.27)$$

$$\ln(s_{w6}^{ind}) \approx -0.51 \ln(1 - Q_{w6}) - 2.37 \quad (4.28)$$

$$\ln(s_{w9}^{ind}) \approx -0.52 \ln(1 - Q_{w9}) - 1.94 \quad (4.29)$$

$$s_{w3}^{ind} \approx (1 - Q_{w3})^{-0.51} e^{-2.48} \quad (4.30)$$

$$s_{w6}^{ind} \approx (1 - Q_{w6})^{-0.51} e^{-2.37} \quad (4.31)$$

$$s_{w9}^{ind} \approx (1 - Q_{w9})^{-0.52} e^{-1.94} \quad (4.32)$$

$$\ln(s_{w3}^{snap}) \approx -0.33 \ln(1 - Q_{w3}) - 2.06 \quad (4.33)$$

$$\ln(s_{w6}^{snap}) \approx -0.33 \ln(1 - Q_{w6}) - 2.01 \quad (4.34)$$

$$\ln(s_{w9}^{snap}) \approx -0.34 \ln(1 - Q_{w9}) - 1.86 \quad (4.35)$$

$$s_{w3}^{snap} \approx (1 - Q_{w3})^{-0.33} e^{-2.06} \quad (4.36)$$

$$s_{w6}^{snap} \approx (1 - Q_{w6})^{-0.33} e^{-2.01} \quad (4.37)$$

$$s_{w9}^{snap} \approx (1 - Q_{w9})^{-0.34} e^{-1.86} \quad (4.38)$$

422 Chapter 5

423 Starshow

424 This chapter will discuss the Starshow application. Starshow is a visualizer for
425 star dynamics simulations. It enables users to navigate through the simulated
426 particles, visualize their trajectories and read out their data.

427 There already exist various applications for the visualization of star dynam-
428 ics simulations. These applications however, allow for only a limited amount
429 of interaction since they are restricted to the input and output offered by a
430 standard desktop pc. Figure 5.1 shows a picture of the platform starshow was
431 build for, the Personal Space Station [10] (PSS) used at the UvA SCS group.
432 With the creation of Starshow for the Personal Space Station we intended to
433 create a visualizer for star dynamics simulations that allowed for new sorts of
434 interactions and more meaningful feedback for the astrophysicist.

435 Besides being innovative, we wanted Starshow to be an application that as-
436 trophysicists would actually be able to use for research. The Personal Space
437 Station has three properties making it very accessible for normal researchers.
438 The Personal Space Station is very affordable, thus there is a reasonable pos-
439 sibility that a researcher can persuade his institute to purchase such a device.
440 The size of the Personal Space Station is such that it can fit on a desk, thus a
441 separate desk is all that is needed to be able to place it somewhere. It is made
442 from of the shelf components, thus if something breaks, the institute can replace
443 it by itself.

444 5.1 The Personal Space Station

445 Virtual reality (VR), although there is no official definition for it, usually refers
446 to immersive virtual reality. With immersive virtual reality the user is immersed
447 in a computer generated 3-dimensional environment. Well known examples of
448 fully immersive virtual reality environments are head-mounted displays (HMD)
449 and the CAVE (CAVE Automatic Virtual Environment) [7]. The PSS however
450 does not provide fully immersive virtual reality; instead, it provides a semi-
451 immersive virtual reality, a virtual reality where the user only immerses his
452 hands. The PSS is described by its designers as a near-field virtual environment
453 since it provides a virtual reality that is focused on users interacting with objects
454 within arm's reach.



Figure 5.1: Personal Space Station

455 5.1.1 History

456 The PSS I worked on during this project was built at the UvA SCS group by
457 R. Shulakov [17] who based his design on the work of J.D.Mulder and R. van
458 Liere [10].

459 The first mirror-based, near-field, virtual environment was proposed by C.
460 Schmandt [15] in 1993. His design used a half-silvered mirror for output, al-
461 lowing the user to see their real hands and the computer display as shown in
462 figure 5.2. For positional input he used a 6 degree-of-freedom magnetic tracking
463 device. He did consider tracking by using optical tracking with infrared light,
464 but since obscuration by the body is a significant issue when using infrared

465 light, he decided to use magnetic tracking instead. The problem with magnetic
466 tracking however is that the CRT monitor used for display caused magnetic
467 interference with the tracker.

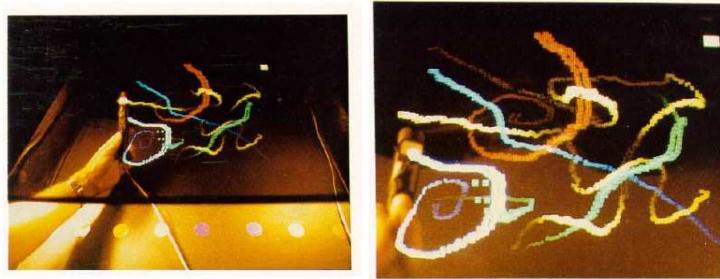


Figure 5.2: The half-way mirror used by Schmandt.

468 When the PSS was being created there were already several designs based
469 on Schmandt's work. In 1994 Poston et al. [18] created the Virtual Workbench,
470 which used mechanical (the Immersive ProbeTM) as well as electromagnetic
471 sensors (the Pholhemus FasTrakTM) for user input.

472 In 1999 Wiegand et al. [19] created their version of the Virtual Workbench
473 which made use of a haptic interface, a mechanical manipulator called PHAN-
474 ToM with a 3 DOF force-feedback and a trackball. There is also a Swedish com-
475 pany called Reachin [6] that sells virtual workbenches that utilizes the PHAN-
476 ToM manipulator for input.

477 The PSS differs in two ways from these systems: first of all, the PSS utilizes
478 head-tracking to be able to display the image correctly and second, the PSS
479 uses optical tracking with infrared lighting for the interactive tasks. None of
480 the previous systems applied these methods.

481 5.1.2 Workings

482 Three spaces

483 Figure 5.3 illustrates the layout of the PSS. The PSS distinguishes 3 spaces: the
484 visual space, the tracking space and the interaction space. The visual space is
485 the virtual space the user can perceive, while the tracking space is the space
486 covered by the cameras. The interaction space is the space where the user can
487 actually interact.

488 The visual focus plane (VFP) is the reflected image of the monitor in the
489 mirror. According to Mulder and van Liere, due to accommodation and conver-
490 gence conflicts, the useful depth range of the visual space is limited to $+/- 10$
491 centimeters around the focus plane.

492 Since the interaction space is restricted to where the user can reach it is
493 important to configure the VFP and tracking space in such a way that their
494 intersection will completely overlap the user's interaction space.

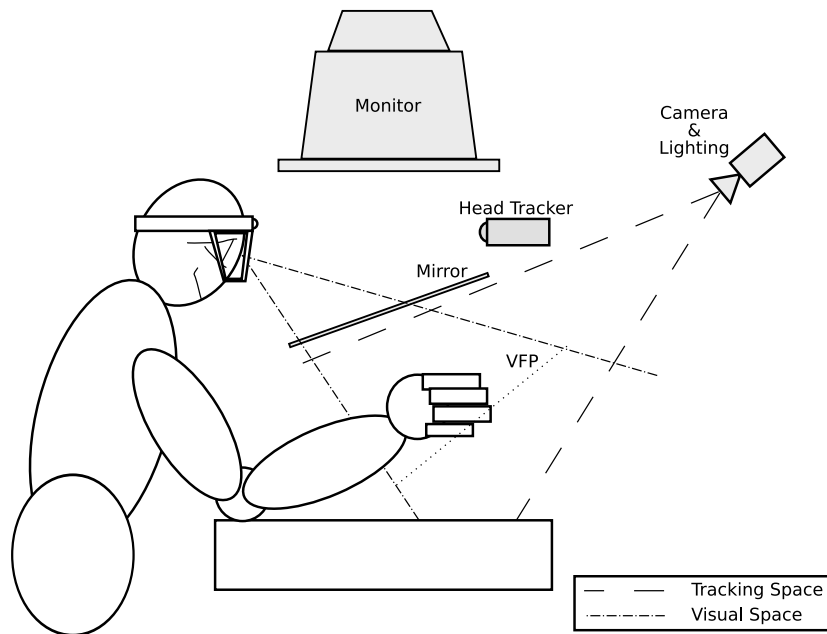


Figure 5.3: Layout of the PSS.

495 Interaction

496 The PSS utilizes optical tracking with infrared lighting to track objects in the
 497 tracking space. This method is described in [13]; I will give a short summary
 498 here.

499 Each object used for interaction with the PSS contains a number of retro-
 500 reflective markers. These markers are ordered in several patterns enabling the
 501 PSS to determine the location and orientation of the object. For the PSS to
 502 be able to do this it needs to do 3 things: train, recognize the patterns and
 503 determine the location and orientation of these patterns.

504 The PSS needs to be only trained once for each interaction object. During
 505 this training session the PSS stores several properties of the patterns he detects
 506 in a database. The properties being stored are invariant under perspective
 507 projections and are therefore well suited for this problem.

508 During normal operation the PSS uses the stored projective invariant prop-
 509 erties to match the detected markers to the patterns in his database. Once he
 510 has found the matching pattern he can determine its orientation and thus the
 511 orientation of the interaction object.

512 Specifications

513 The PSS installation I worked on makes use of dual 2.4 GHz Xeon processors
 514 and 2 Gb of memory. The graphics card is of the NVIDIA Quadro4 900 XGLtm
 515 brand, providing quadro buffered stereo rendering. The shutter glasses are
 516 of the Crystal Eyestm brand and work in combination with the Crystal Eyes

517 ESGI Emitter to synchronize with the monitor. Headtracking is done using the
518 DynaSighttm optical radar by Origin instruments. There are 4 B/W, 25 fps,
519 PAL resolution, interlaced cameras used for the hand-tracking in combination
520 with a Leutron PicPort Stereo H4 framegrabber. The total costs of the PSS are
521 estimated to be around €15.000,-.

5.2 The Idea behind Starshow

523 The reason that Starshow is developed with the PSS in mind was that a regular
524 desktop visualization provides too little freedom for comfortable interaction with
525 starclusters consisting of a large number of stars. By using the PSS the user
526 gets extra feedback in the sense of the position/angle of his arms and hands to
527 help him navigate through and interact with the star clusters.

528 Starshow has been implemented entirely in OpenGL and uses the VRCO
529 CaveLib, therefor ensuring that it will not only run on the PSS, but in a regular
530 CAVE environment as well. Currently it supports only the starlab format;
531 however, its internal data structure and I/O system is independent of that of
532 Starlab so that it is capable to support multiple file-formats in the future.

533 Besides being able to run on the PSS, we also wanted the application to be
534 able to run on the CAVE. This is far more useful if someone wants to give a
535 presentation of some sort to a number of spectators. The reason that it has
536 been designed primarily for the PSS is that although the CAVE environment is
537 very nice, it is hardly used for everyday research. The reason for that is that
538 it is actually too big to really work with; there is no possibility to put a CAVE
539 in your office. On top of that it is also very expensive, resulting in a lot of
540 research institutes having only one or even none at their facility; thus even if a
541 researcher's institute has a CAVE, that researcher would need to reserve it if he
542 wanted to use it.

5.3 Interaction

544 Starshow supports 4 types of PSS controls: the wand, the cube, the palette and
545 the joystick. The wand and joystick are also available in the CAVE environment;
546 the cube and palette however are only available for the PSS environment.

547 Although the wand is available for the PSS as well as the CAVE environment,
548 it differs quite significantly for both systems. The wand for the PSS has the
549 shape of a pen and it's only capability for interaction is being able to point and
550 3 buttons. The wand in the CAVE environment has the shape of a StarTrek
551 phaser and has just as the PSS version 3 buttons. However, next to those
552 buttons it has a joystick integrated into it.

553 Each PSS control has its own functionality:

554 **Wand** The wand, shown in figure 5.4, has 3 buttons and usually acts as a sort
555 of mouse in the PSS environment. In Starshow its function is threefold: it is
556 used to give direction when navigating, to select stars and to navigate through
557 menus.

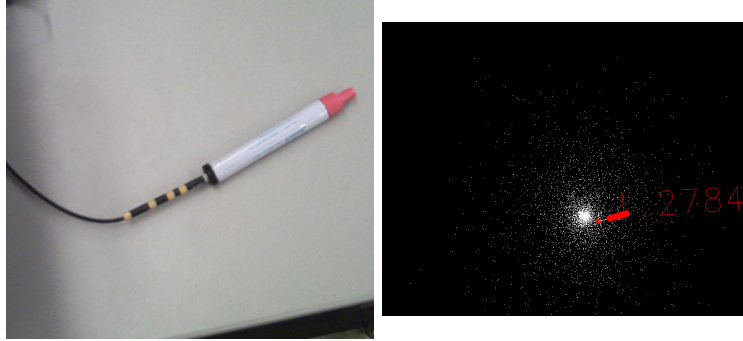


Figure 5.4: The wand. The left image shows the physical device and the right image how it is represented in Starshow.

558 **Joystick** The Joystick is just a regular joystick from Logitechtm (Logitech
 559 Extreme Digital 3D). It is used to control the navigation speed and to be able to
 560 navigate sideways. Because the Sidewinder has a numerous amounts of buttons
 561 it is also used to activate several modes in the application.

562 **Cube** The cube, depicted in figure 5.5, is used for rotating the navigation
 563 space in the application. However due to the fact that there is no real ‘center’
 564 to rotate around, especially when visualizing multiple globular clusters, this is
 565 not a really useful feature, since it is more likely to confuse the user than to
 566 assist him.

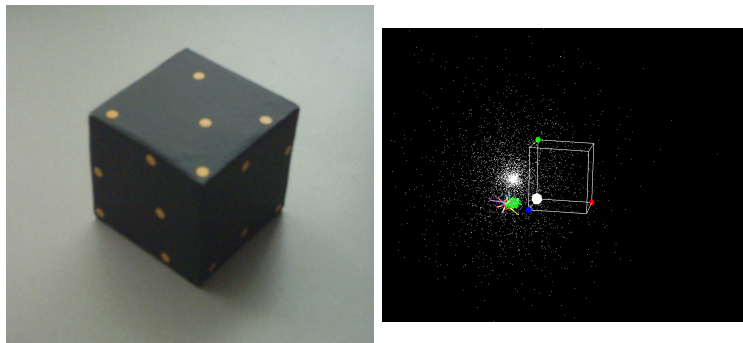


Figure 5.5: The cube. The left image shows the physical device and the right image how it is represented in Starshow.

567 **Palette** The palette can be considered an extended version of the cube. It is a
 568 board with a cube attached to it at the top left for tracking purposes as can be
 569 seen in figure 5.6. The palette is used for the display of menu and miscellaneous
 570 data.

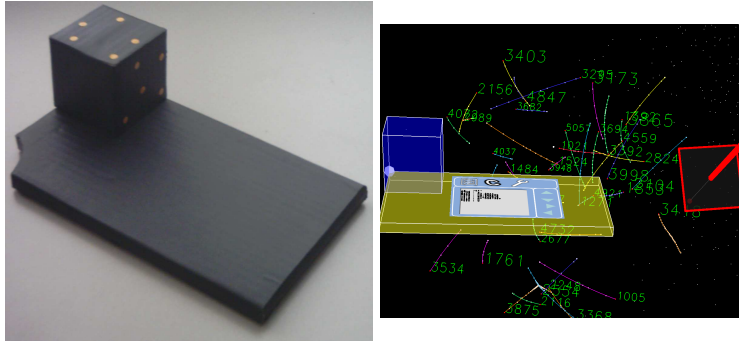


Figure 5.6: The pallet. The left image shows the physical device and the right image how it is represented in Starshow.

571 5.4 Selecting Stars

572 The most important element for any interactive environment that uses a point-
 573 ing device is the ability to select. This means that a user can point the pointing
 574 device at the item he wishes to select and use it. When selecting items in menus,
 575 this is not a very big issue; however when you have a large amount of particles
 576 each depicted as a point located somewhere in a large space, this becomes a bit
 577 trickier.

578 5.4.1 Selection mechanism

579 Since the stars are visualized using perspective projection, I decided to use
 580 a cone shaped selection volume method for the selection of the stars. When
 581 perspective projection is used and a point at the screen is chosen to represent
 582 the projection of a line orthogonal to the screen, several stars may appear as if
 583 they are located at the same distance from that line while in reality they aren't;
 584 Since the user in fact actually wants to select those stars that *appear* to be at a
 585 certain distance from that imaginary line which is smaller than a given value,
 586 the cone shaped selection volume is ideal for selection.

587 To be able to explain better how I determine whether a star falls within
 588 the selection cone I will explain some basic linear algebra first; all these prin-
 589 ciples can be found in basic linear algebra literature. I will assume some basic
 590 principles concerning linear algebra are known by the reader however.

591 Inproduct

The inproduct (or dot product) of two vectors \vec{v} and \vec{w} in \mathbb{R}^3 is noted as $\langle v, w \rangle$ and is defined as:

$$\langle v, w \rangle = v_1w_1 + v_2w_2 + v_3w_3 \quad (5.1)$$

592 The inproduct $\langle v, w \rangle$ in \mathbb{R}^3 has the following properties:

- 593 • $\langle v, w \rangle = \langle w, v \rangle$ (commutative)

- 594 • $\langle u + v, w \rangle = \langle u, w \rangle + \langle v, w \rangle$ (distributive)
- 595 • $\langle \lambda v, w \rangle = \langle v, \lambda w \rangle = \lambda \langle v, w \rangle$ (linearity property)

596 The inproduct between two vectors gives an indication of the type of angle
 597 between the two vectors:

- 598 • $\langle v, w \rangle > 0$ if the angle is acute (i.e. $[0, \frac{\pi}{2})$)
- 599 • $\langle v, w \rangle < 0$ if the angle is obtuse (i.e. $[\frac{\pi}{2}, \pi)$)
- 600 • $\langle v, w \rangle = 0$ if they are perpendicular (i.e. an angle of $\frac{\pi}{2}$)

Since the length of a vector \vec{v} is defined as:

$$\|v\| = \sqrt{v_1^2 + v_2^2 + v_3^2} \quad (5.2)$$

601 the inproduct of $\langle v, v \rangle$ yields $\|v\|^2$.

602 Projection

The projection of a vector \vec{v} onto vector \vec{w} is noted as $\pi_w(v)$ and is defined as:

$$\pi_w(v) = \frac{\langle w, v \rangle}{\langle w, w \rangle} w = \frac{\langle w, v \rangle}{\|w\|^2} w \quad (5.3)$$

603 The selection method

604 Figure 5.7 shows the representation of a wand pointing in the direction of a star
 605 including the selection cone. Vector \vec{w} is the normalized direction vector of the
 606 wand, vector \vec{P}_w the positional vector of the wand and \vec{P}_s the positional vector
 607 of the star. The goal of selection is to determine if a star is inside that cone;
 608 thus to determine if $\|\vec{d}\|$ is smaller than $\|\vec{r}\|$ times $\|\vec{l}\|$.

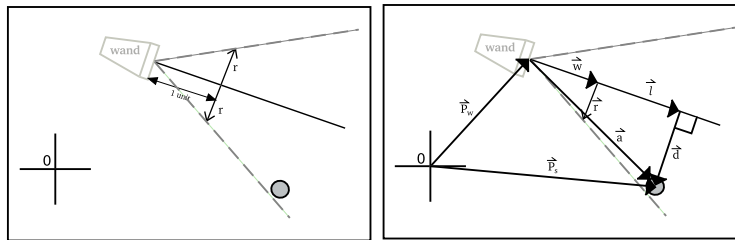


Figure 5.7: The selection cone. On the left the general idea. On the right the values that are used to determine if a star falls into the cone.

First we determine the vector \vec{d} :

$$\begin{aligned} a &= P_s - P_w \\ d &= a - l \end{aligned} \quad (5.4)$$

The vector \vec{l} is the projection of \vec{a} onto \vec{w} :

$$l = \pi_w(a) = \frac{\langle a, w \rangle}{\langle w, w \rangle} w$$

$$d = a - \frac{\langle a, w \rangle}{\langle w, w \rangle} w \quad (5.5)$$

609

Because $\langle w, w \rangle = \|w\|^2$ and vector \vec{w} has length 1 because it is normalized, it means 5.5 can be simplified to

$$l = \langle a, w \rangle w$$

$$d = a - \langle a, w \rangle w \quad (5.6)$$

610

All stars that are within the selection cone should meet the following requirement:

$$\|d\| < \|l\| \|r\| \quad (5.7)$$

But this can be reduced in the following way:

$$\|d\| < \|l\| \|r\|$$

$$\|d\| < \|\langle a, w \rangle w\| \|r\|$$

$$\|d\| < |\langle a, w \rangle| \|w\| \|r\|$$

$$\|d\| < |\langle a, w \rangle| \|r\| \quad (5.8)$$

611

This then eventually results in:

$$\|d\|^2 < \langle a, w \rangle^2 \|w\|^2 \|r\|^2$$

$$\langle d, d \rangle < \langle a, w \rangle^2 \langle r, r \rangle \quad (5.9)$$

$$\langle d, d \rangle < \langle P_s - P_w, w \rangle^2 \langle r, r \rangle \quad (5.10)$$

612 5.4.2 Preliminary test

613 Because you don't want to run the complete test for every star, even those
 614 that are behind the user, a preliminary test is desired. A very simple test to
 615 eliminate these stars is by testing the value of their inproduct $\langle a, w \rangle$; if this
 616 value is negative, then that means the angle between vectors \vec{a} and \vec{w} is obtuse,
 617 which means the star is behind the wand.

618 The algorithm

619 When clicking the select button while pointing your wand in an environment
 620 with a cluster of several thousand stars the following algorithm is executed:

```

621 constant : pos_wand, w, selection_range, pos_star(n)
622
623 for star i= 1:n
624 {
```

```

625     a = vec_pos_star(i) - vec_pos_wand;
626
627     aw = dot ( a, w);
628
629     if ( aw > 0 )
630     {
631         d = vec_a - aw * vec_w;
632
633         if ( dot( d,d ) < aw * aw * selection_range )
634         {
635             toggle star(i) as selected;
636         }
637     }
638 }

```

639 The complexity of the algorithm in big O notation is $O(n)$, meaning that
640 the complexity is linear, i.e. if you increase the number of stars in the system a
641 times, it means that the number of computations will increase no more than a
642 times accordingly.

643 The processing required by this algorithm is easily determined. Assuming
644 there are a total of n stars and m of these are in the users view during selection
645 the number of processing $p_t(m, n)$ required is:

$$p_t(m, n) = n(5p_a + 3p_m + p_c) + m(5p_a + 8p_m + 1p_c) \quad (5.11)$$

646 where p_a stands for the processing required for additions and subtractions,
647 p_c for conditional checks and p_m for multiplications. It is well known that the
648 addition, subtraction and conditional operations require far less processing than
649 multiplications. This means that when the number of particles is large enough,
650 it will be the multiplications that will require practically all the processing time
651 for the selection; and when selection is performed with all the particles in front
652 of the wand, the required processing can be simplified to:

$$p_t(n) = n11p_m \quad (5.12)$$

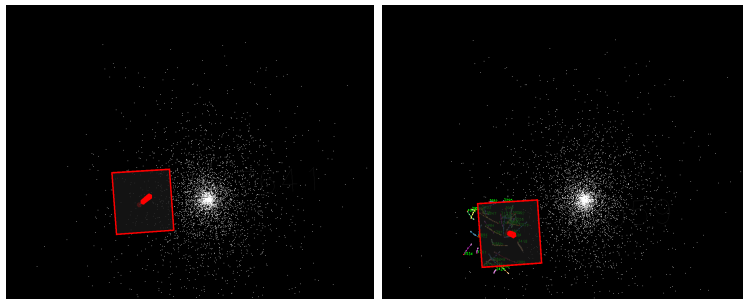


Figure 5.8: Wand in selection mode. Figure 5.9: Selecting several stars.

653 5.5 Visualizing Stars

654 Star simulations representing clusters of a huge number of stars are not un-
655 common, thus it is important to minimize the processing required per star,
656 especially considering it is an interactive application. Each star is therefore vi-
657 sualized as a single point, even though this does make depth perception for the
658 user somewhat difficult.

659 5.5.1 Visualizing stars over time

660 To be able to display a frame containing stars, the stars' positions will first need
661 to be interpolated to a given time instance.

662 The way starshow does this is as follows: first the time instance t is deter-
663 mined; this is the time instance for which the stars need to be displayed. Then,
664 for every star, the star instance is searched with the largest time instance t_{star}
665 that is still smaller than t . This search is started from the star instance used
666 for the previous interpolation. The selected star instance and the star instance
667 that follows directly in time are then used for interpolation.

668 In pseudo code:

```
669  
670 t = get_current_time()  
671  
672 for ( i=0; i < N; i++ )  
673 {  
674     curStarInstance = lastSelectedInstance[ i ];  
675  
676     if ( t < curStarInstance->time )  
677     {  
678         curStarInstance = Stars[ i ]->timeLine->first  
679     }  
680  
681     while ( curStarInstance->next->time < t )  
682     {  
683         curStarInstance = curStarInstance->next  
684     }  
685  
686     selectedInstance = curStarInstance;  
687  
688     interpolatedStars[ i ] =interpolate( selectedInstance, selectedInstance->next, t )  
689  
690     lastSelectedInstance[ i ] = selectedInstance;  
691 }  
692
```

693 where N is the number of stars in the model being displayed.

694 Trajectory

695 The time instance from the simulation that is currently being visualized will
696 be referred to as the current simulation time. When a user selects a star, its

697 trajectory from the time instance at which the simulation started till the current
698 simulation time will be shown.

699 This trajectory is constructed by connecting all the star's instances from
700 the beginning of the simulation up to the latest instance with a time instance
701 smaller than the current simulation time; this latest instance is connected to the
702 current interpolated position of the star. Each pair of instances is connected by
703 a set of $n + 1$ straight lines through n points; the positions of these points are
704 determined by Hermite interpolation as described in section 5.5.1. The position
705 for the i^{th} point is determined by interpolating the position of the two instances
706 for the time instance $\Delta t \frac{i}{n+1}$, where Δt is the difference in time between the two
707 instances.

708 A larger n requires more processing power, while providing a better trajec-
709 tory. Therefore Starshow allows the size of n to be specified by the user, thus
710 allowing the user to scale the required processing data for the trajectories when
711 needed. The default size of n is 0.

712 Binaries

713 A very interesting class of stars are the binary stars. A binary star is a pair
714 of stars bound together by gravity, orbiting around their 'center of mass'. The
715 velocity of a star that is part of a binary, can be divided into two components:
716 the velocity of the center of mass and the relative velocity of a member of the
717 binary star with respect to the center of mass. The velocity of the center of
718 mass can be regarded as the velocity of the entire binary star, while the relative
719 velocity indicates the velocity at which the members of the binary rotate around
720 the center of mass.

721 Although the rotational velocity of a member of a binary star might be
722 very high, while rotating around the center of mass, the distance of a member
723 of a binary star from the center of mass doesn't change noticeably. Since the
724 rotational velocity is usually several factors larger than the velocity of the center
725 of mass, the absolute velocity of a member of a binary star is mainly determined
726 by this relative velocity.

727 Figure 5.10 shows an illustration of the effects of using either velocity. In
728 image A, the absolute velocity is used for interpolation and the image clearly
729 shows that the interpolated trajectory deviates extremely from what its actual
730 trajectory might be. Image B shows the effect of using the velocity from the
731 center of mass; this interpolation also clearly deviates from its actual trajectory,
732 however it stays within the bounds formed by the position of the center of mass
733 and the member's distance to the center of mass.

734 Therefore, when interpolating the position of a member of a binary we use
735 the velocity of the center of mass instead of its own absolute velocity. Figure 5.11
736 shows the trajectories of the members of a binary star as they are interpolated
737 by Starshow.

738 Hermite Interpolation

739 Starshow uses a very basic interpolation scheme to interpolate the position of
740 particles called Hermite interpolation [8]¹. This interpolation is specifically

¹cubic hermite interpolation to be more precise

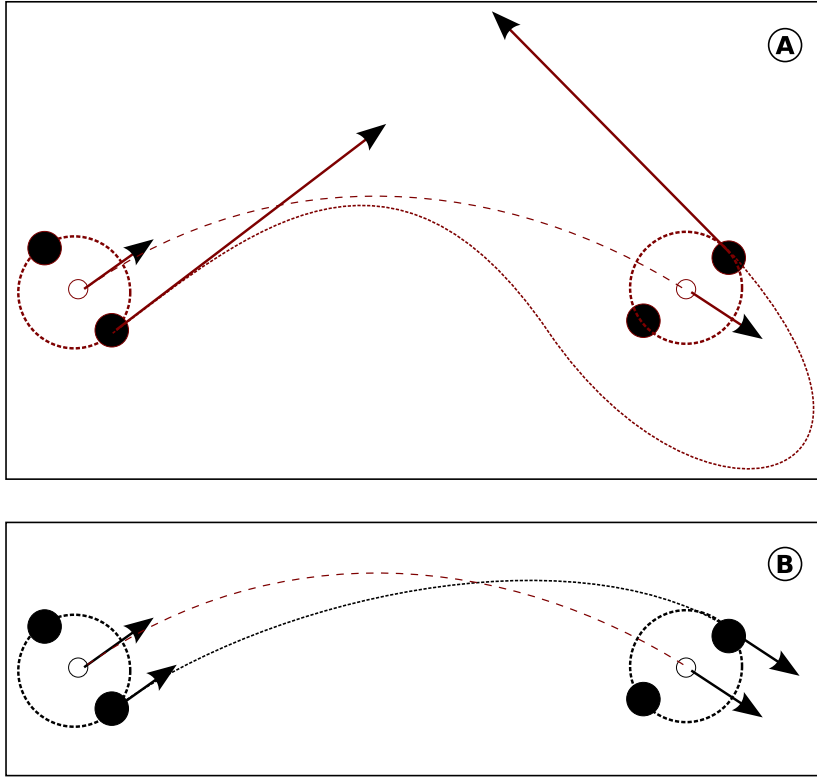


Figure 5.10: Image (A) shows the effect when interpolating the position of a member of a binary using its own absolute velocity; Image(B) shows the effect when interpolating the position of a member of a binary using the velocity of the ‘center of mass’ of that binary.

741 designed for the interpolation of a curve defined by two end-points with each a
 742 certain tangent.

743 Given two end-points \vec{P}_1 and \vec{P}_4 with tangents \vec{R}_1 and \vec{R}_4 respectively, the
 744 cubic Hermite interpolation is defined as equation 5.13.

$$\vec{Q}(t) = (2t^3 - 3t^2 + 1)\vec{P}_1 + (-2t^3 + 3t^2)\vec{P}_4 + (t^3 - 2t^2 + t)\vec{R}_1 + (t^3 - t^2)\vec{R}_4 \quad (5.13)$$

745 The resulting $\vec{Q}(t)$ is the interpolated position at the given time t . However,
 746 there is one constraint, the given time t has to be in the interval $0 \leq t \leq 1$,
 747 with $\vec{Q}(0) = \vec{P}_1$ and $\vec{Q}(1) = \vec{P}_4$. Since the time intervals between star instances
 748 are not so neatly constrained, the simulation time needs to be scaled to fit the
 749 constraints.

The scaling is done as follows:

$$t = \frac{t_{curr} - t_1}{t_4 - t_1} \quad (5.14)$$

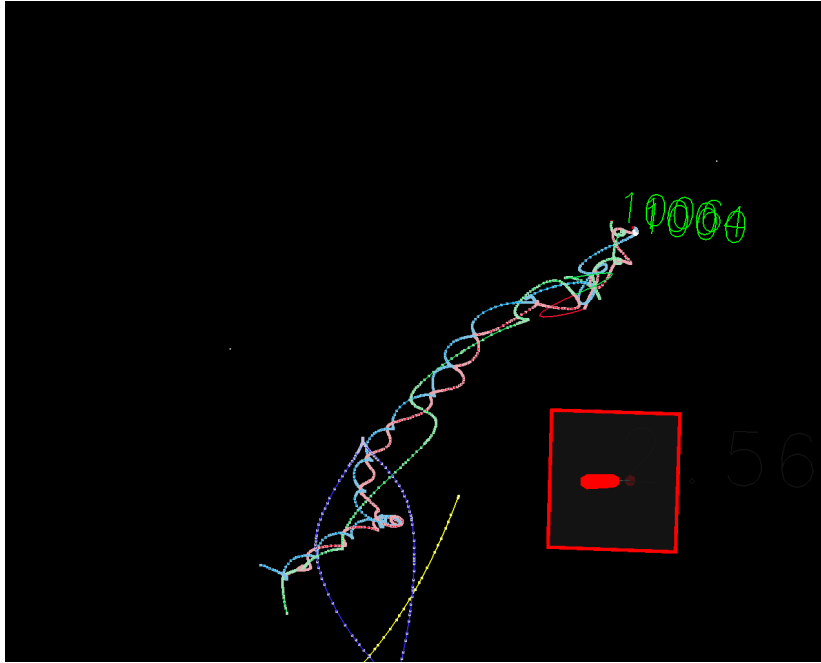


Figure 5.11: Several particles forming binaries.

750 Time t_{curr} stands for the current simulation time and times t_1 and t_4 for the
 751 times at the begin and end points respectively.

752 But now that the time has been scaled, the tangents of the end-points need
 753 to be adjusted accordingly. Therefore a small adjustment has to be made in the
 754 Hermite interpolation resulting in:

$$\vec{Q}(t) = (2t^3 - 3t^2 + 1)\vec{P}_1 + (-2t^3 + 3t^2)\vec{P}_4 + (t_4 - t_1)((t^3 - 2t^2 + t)\vec{R}_1 + (t^3 - t^2)\vec{R}_4) \quad (5.15)$$

755 5.5.2 Multi-threading

756 When the system being used for visualization contains a great deal of particles,
 757 interpolating the positions of these particles can require quite some time. When
 758 this happens and the program is single threaded, the user will have to wait till
 759 all interpolation is done for each cycle until he can interact with the program
 760 (for a very short time). This, of course, does not improve the user experience.
 761 To prevent this from happening we use ‘multi-threading’ and ‘buffering’.

762 Threading is a technique that allows a program to split itself into two or more
 763 simultaneously running tasks. Thus allowing to interpolate the positions of a
 764 collection of particles, while simultaneously still being able to receive user input.
 765 Figure 5.12 is an UML activity diagram of Starshow, illustrating how the pro-
 766 gram is split up in (at least) two threads after `CAVEInit()` is called. The main

767 process regularly updates the model by calling the function `displayUpdate()`
768 . The display thread however continuously reads user input and displays the
769 model last interpolated model.

770 5.5.3 Buffering

771 The display thread displays the model while simultaneously the model is being
772 interpolated; to prevent these two threads from interfering with each other's
773 data, buffers are needed.

774 Starshow uses two buffers and two pointers to these buffers which I will refer
775 to as pointer A and B. The buffer from pointer A contains already interpolated
776 data. This data is used to display and this is what the user is navigating through.
777 The buffer from pointer B is being written to by the function performing the
778 interpolation of the positions of the particles.

779 When the interpolation of the model is completed, the pointers will be
780 switched; pointer A will point to the buffer of pointer B and vice versa.

781 This way, the user will always be able to navigate smoothly, even if the
782 interpolation requires a lot of processing; when the interpolation of the model
783 requires a lot of processing, the user will only notice stars standing still or
784 moving sluggish, but will not be hampered in hist navigation.

785 5.6 Menus

786 There are two types of menus available in Starshow; both present the same op-
787 tions and data, however the ways they are presented to the user are significantly
788 different.

789 5.6.1 Floating Menu

790 The first and more classic menu type is the *floating* menu, shown in figure 5.13.
791 This menu floats in space at a fixed coordinated relative from the users point of
792 view. The user chooses an item from the menu by pointing at the desired item
793 using the Wand and selecting it by clicking a button on the Wand. Because it
794 is very hard to navigate through a floating menu in a PSS environment, it is
795 hardly utilized in Starshow.

796 5.6.2 Palette Menu

797 The second and relatively uncommon type of menu is the *palette-menu* 5.14.
798 This menu is displayed in the virtual space, superimposed over the Palette as
799 shown in figure 5.14.

800 The user is capable of selecting an item by tapping on the palette using the
801 Wand. Since the PSS is not always correctly configured, causing the user to be
802 actually unable to tap the palette in virtual space; therefore the user is given
803 an alternative method of selecting options by clicking the first button on the
804 Wand.

805 Since the user is actually holding the menu, he has full control over the
806 position of the menu. This adds another level of interactivity and lends itself

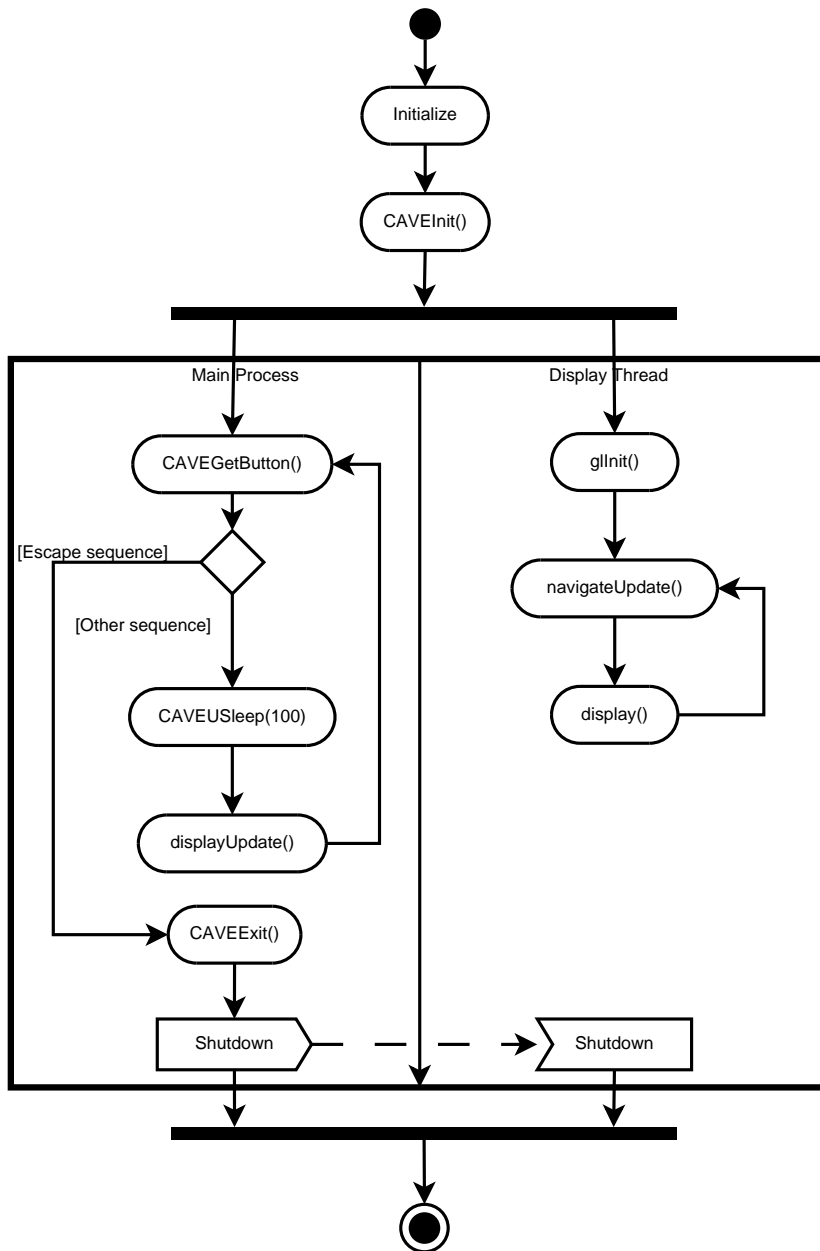


Figure 5.12: UML activity diagram illustrating Starshow's multi-threading.

807 for the display of specific data about the object displayed and selected in the
 808 virtual space.

809 By making the user tap the palette to choose and select a menu item, the

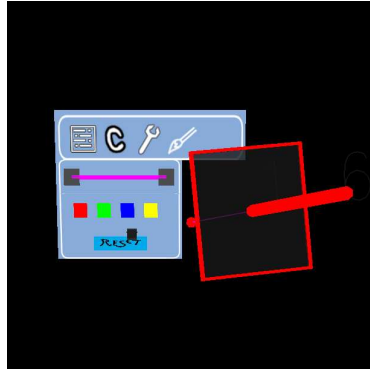


Figure 5.13: The floating menu.

810 application provides the user with extra feedback, creating an effect of really
 811 ‘touching’ objects in VR. Since it is impossible for the palette-menu to be
 812 realized in the CAVE-environment, the palette menu will only be available when
 813 using Starshow in the PSS environment.

814 5.6.3 Data window

815 The data window in the palette menu shows the statistics for the currently
 816 focused star instance, e.g. id, time, position, velocity etc.

817 By selecting the up and down arrows, the user can make the focus traverse
 818 the currently selected star’s trajectory through time, as illustrated in figure 5.14;
 819 the data window will show the data for the currently focused star instance on
 820 that specific trajectory.

821 Using the left and right arrows, the user can make the focus traverse through
 822 a bundle of selected stars. This is demonstrated in figure 5.15; in the top image,
 823 the focus is on star #1021 from the bundle of selected stars and by using the
 824 palette menu this focus is changed to star #1156 in the bottom image.

825 5.7 Starshow’s Internal Data Structure

826 One of the requirements was that Starshow should not only be able work with
 827 the starlab data format, but also with other data formats. At this moment
 828 however it only supports the Starlab data format and uses the libraries provided
 829 by Starlab to be able to read and parse output generated by Starlab. Besides the
 830 parsing functionality the Starlab libraries also offer classes for storing the stellar
 831 information. Using these classes would make Starshow completely dependent
 832 on Starlab, therefore they are not utilized by Starshow. Though Starshow has
 833 some dependencies on Starlab, we tried to keep it at a minimum.

834 The basic structure of the classes used to store the particle data is shown in
 835 the UML class diagram in figure 5.16. It illustrates the relations between the 5
 836 main classes: StarInstance, StarContainer, TimeListNode, TimeLine and Star.

837 **StarInstance** The StarInstance class contains all the properties for a star for
838 a given time instance of that star such as its position, velocity, acceleration,
839 time instance, mass etc.

840 **StarContainer** The StarContainer class contains one StarInstance. It also
841 contains a list of daughters and a link to a parent StarContainer. The StarCon-
842 tainer class represents a node in the tree of the input data.

843 **TimeLine and TimeListNode** The TimeLine of a Star is a linked list rep-
844 resenting the timeline of that star. The nodes in the TimeLine list are TimeLi-
845 neNode classes, each containing one StarContainer instance.

846 **Star** The Star class contains one TimeLine object and the global data of the
847 star it represents, such as its identifier, selection flag etc.

848 All Star objects for actual stars (no Center of Mass particles or other abstract
849 objects) are put in one large global array called the StarArray. The reason for
850 choosing an array instead of a linked list is that it is much faster to select a
851 certain particle from an array than from a linked list. Since the set of Stars
852 is created only once and is not restructured, there is no need to worry about
853 rebuilding or adding particles to the list. An array is therefore quite practical.

854 All Star objects that are selected by the user will be added to the Selected-
855 StarList linked list. This is a linked list since the list containing the ‘currently
856 selected stars’ is very dynamic.

857 5.8 Used Libraries

858 5.8.1 CAVELib

859 The PSS on which Starshow was developed, made use of the CAVELib library.
860 This library provides an API designed by VRCO [2] for developing software
861 for immersive displays. The same API is being used for visualization in the
862 CAVE. This has the advantage that when an application is being designed for
863 the PSS, it should work in the CAVE as well. However, the PSS only uses a
864 single screen, while the CAVE uses multiple, therefore applications designed for
865 the PSS usually require some modifications before they are capable of running
866 on the CAVE.

867 Another difference between the CAVE and the PSS is, since the PSS’ in-
868 teraction principle is so significantly different from that of the CAVE, certain
869 types of applications made for the PSS have no use in the CAVE environment
870 altogether.

871 5.8.2 OpenGL

872 For the actual visualization OpenGL [1] was used. OpenGL stands for Open
873 Graphics Library and was originally designed by SGI. It is a standard specifi-
874 cation defining a cross-platform API for 2D and 3D computer graphics. The
875 choice for using OpenGL was simple, since there is no real alternative for it on
876 non-Windows platforms.

877 5.8.3 Starlab

878 The Starlab libraries are used by the Starshow application for parsing Starlab
879 data files and nothing more, because we needed to be able to parse Starlab
880 data. Using the Starlab libraries would require a user to have Starlab compiled
881 on his system before he can compile Starshow, therefore I preferred parsing the
882 Starlab data files without the use of the Starlab libraries. However, due to a
883 lack of documentation and knowledge available concerning the data format I
884 was unable to parse it myself correctly, leaving no other choice than using the
885 Starlab libraries.

886 5.8.4 STL

887 The Standard Template Library [16] (STL) is a C++ library for container
888 classes, algorithms, and iterators. The reason for using this library is that
889 it has highly optimized sorting and searching functions, which are necessary for
890 the large amounts of particle data that need to be parsed and loaded into mem-
891 ory. That is also the only time that this library is used; during the parsing and
892 loading of the particle data. Because the memory management in CAVELib
893 causes difficulties with the STL containers, these containers are only used when
894 the CAVELib memory management is not active, which is at the beginning of
895 the application during the loading and parsing of input files. After the data
896 has been parsed, loaded and sorted into the STL containers it is moved into
897 Starshow's own classes.

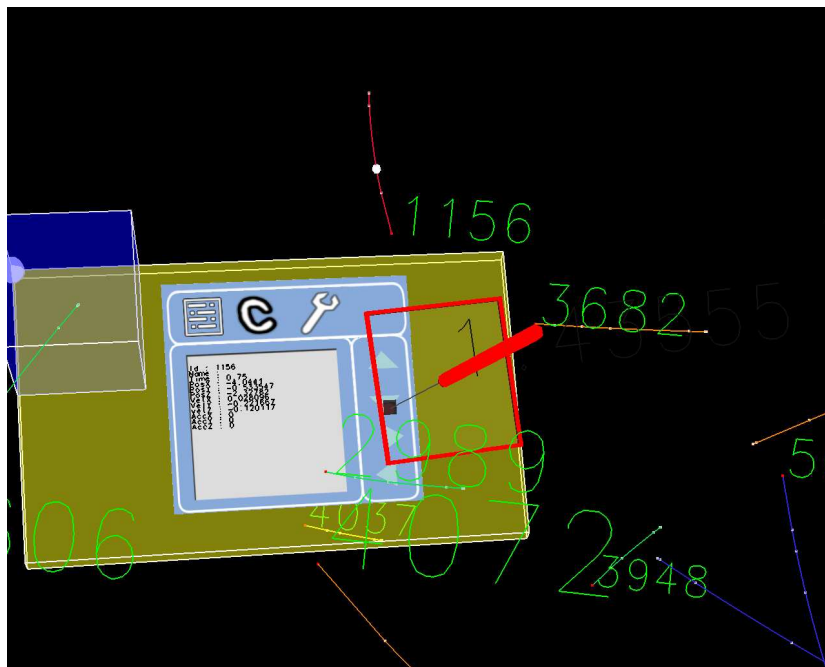
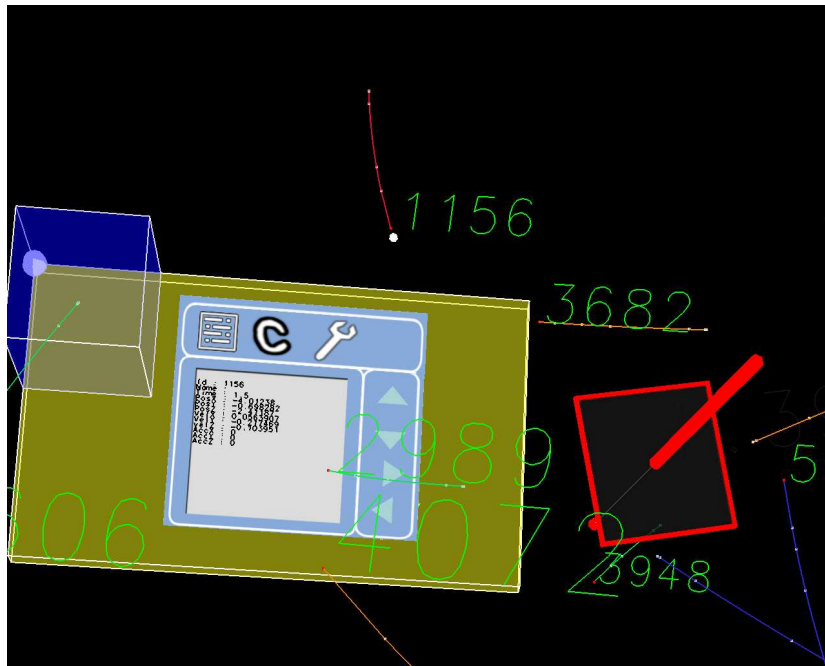


Figure 5.14: The user is capable of backtracking a star over its trajectory. The top image shows that the star #1156 is selected, and the bottom image how the particle is backtracked for 2 steps.

898 Chapter 6

899 Discussion and Conclusion

900 6.1 Discussion

901 6.1.1 Sampling Schemes

902 Snapshot sampling and individual sampling use different units for their sam-
903 pling intervals; therefore comparing the two sampling schemes directly was not
904 possible.

905 Energy

906 In chapter 4 it was shown that the error in the energy of the individually sampled
907 data can be fitted to a power law function with a power of about 2. However, the
908 energy error quickly approaches the actual energy of the interpolated system,
909 which is -0.25 , and therefore already becomes unreasonably large when the
910 sample interval becomes $> x128$. Data using such a sample interval is therefore
911 unsuited to use as input for a star dynamics integrator.

912 Size & Quality

913 In chapter 4 I showed that the size of the resulting data set created by snapshot
914 sampling can always be determined in advance. The size of a data set created
915 by individual sampling however, cannot be determined in advance because it de-
916 pends on the total number of integration steps of all particles over the complete
917 integration, which is unknown by the user in advance.

918 It was shown that the individual sampling scheme has a far better size/quality
919 ratio than the snapshot sampling scheme. Thus when two data sets, one created
920 with individual sampling and the other with snapshot sampling, have the same
921 size, the quality of the former data set will always be higher than the latter.

922 It showed that an increase of the quality of sampled data, by decreasing $1-Q$
923 with a factor of 10 would result in an increase of the size of snapshot sampled
924 data with a factor of approximately 3.2, while the same increase in quality would
925 result in an increase in size with a factor of about 2.1 for individually sampled
926 data.

927 Even though the tests in chapter 4 were performed using only King models
928 there is no reason to assume the results should be any different if other star

929 cluster models had been used. Even for a system that contains a great amount
930 of chaotic particles, forcing the integrator to use a very small timestep for most
931 of the particles throughout the integration, this would still hold; the resulting
932 dataset produced by individual sampling might be very large, but its quality
933 will still be higher than a dataset of the same size that is generated by snapshot
934 sampling.

935 6.1.2 Starshow

936 Starshow is currently only capable of loading data sets that are formatted using
937 the Starlab format. It is possible to load data generated using individual
938 sampling as well as data sets that are generated using snapshot sampling.

939 When using individually sampled data, trajectories have a higher concentration
940 of data points at places where the trajectory fluctuates than at the places
941 where the trajectory is quite stable; this results in smooth curves when the tra-
942 jectories of stars are shown, even when the number of interpolated points used
943 to draw the lines with, is quite low. The data points of trajectories based on
944 snapshot sampled data on the other hand, are equally distributed over time,
945 resulting in trajectories that deviate much more from the original.

946 Kira's individual timestep integration is what causes the data to be more
947 concentrated when the trajectory fluctuates; however, the fact that Starshow
948 is capable of reproducing this distribution is due to the individual sampling
949 scheme.

950 The advantage of an application such as Starshow over a 'regular' desktop
951 visualization application is that it allows the user to interact more direct with
952 the visualized data. Currently its main use is the user's ability to focus on
953 certain stars and track their data; hereby enabling the user to better investigate
954 the behavior of these stars.

955 Starshow is a nice application, however it probably won't become common
956 use anytime soon. This has mostly to do with the current availability of the PSS;
957 the people that should benefit from Starshow, the astrophysicists, most likely
958 have little desire to go to a specific laboratory where they might find a device
959 such as the PSS (if there is one in the first place) to perform their research.
960 However, due to the relatively low costs involved with building a PSS it is not
961 unreasonable to assume that quite a number of universities can have several of
962 these systems to their disposal in the near future.

963 6.2 Conclusion

964 The individual sampling scheme has the advantage of always resulting in a
965 data set that has a higher quality than a data set of the same size resulting
966 from snapshot sampling; however, it has the disadvantage that its resulting size
967 is unknown in advance. For the snapshot sampling scheme it is the other way
968 around; its advantage is that the resulting size of a data set is known in advance,
969 but its disadvantage is that this data set has a lower quality than a data set of
970 the same size generated by the individual sampling scheme.

971 The individual sampling scheme is a very good sampling scheme, however
972 it is not always the most desired sampling mechanism. It is entirely dependent
973 on the application if this sampling scheme is useful or not. Therefore the user

974 will always need to consider beforehand what type of output he desires. If
975 completely discrete data is what the user actually needs, he should use the
976 snapshot sampling scheme, for it will provide the user with complete static
977 instances of the system at desired intervals; otherwise he should really consider
978 using individual sampled data since it will provide the user with more precise
979 samples requiring far less disk space; when trajectories need to be reconstructed
980 from sampled data the output produced by the individual sampling scheme is
981 far more precise than the output of the same size produced by the snapshot
982 sampling scheme.

983 Another benefit of individual sampling is that in combination with individual
984 block-timestep integration it produces data in relatively small chunks, which
985 could make it very useful for streaming purposes.

986 6.3 Future Work

987 6.3.1 Individual Sampling Scheme

- 988 • Determining the size resulting from the individual sampling scheme for
989 various standard stellar distribution models beside the King model. Once
990 this is known people will be able to make an educated guess about the
991 size of the resulting dataset beforehand for these models as well.
- 992 • Test the individual sampling scheme with other implementations of the
993 individual time-step scheme, e.g. with Aarseth's NBODY code. This
994 could give some insight in how dependent the individual sampling scheme
995 is on the method with which the individual time-steps are determined.
- 996 • Currently, the only way for an application to be able to parse Starlab
997 formatted data files correctly is to use the Starlab libraries; this is due to
998 various odd constructions in the format. Therefore, anyone who wishes
999 to compile an application that is capable of parsing files in the Starlab
1000 format, will need to have the Starlab libraries installed on his system.
1001 However, installing the Starlab libraries on a system requires the user to
1002 have Starlab compiled, which takes a long time and a requires quite a bit
1003 of disk space.
1004 Starlab should either revision its output format or create better docu-
1005 mentation regarding the current format; this way they might increase the
1006 number of applications that support their format and thus increase the
1007 number of contributors for their software package.

1008 6.3.2 Starshow

1009 There is still a lot that can be done for Starshow. Following is a list of possible
1010 improvements:

- 1011 • Implement a slider that enables the user to control the time of the visual-
1012 ization.
- 1013 • Finish the menus. The current implementation of the menus is only a
1014 reference implementation. The only options currently available are the
1015 options regarding star selection.

- 1016 • Support other formats besides that of Starlab, e.g. the NBODY format.
- 1017 • Streaming data input of individually sampled data. This way Starshow
1018 can act as a 'player' for a data file that is located on a remote server in
1019 the same way as for example QuickPlayer acts for a movie-file online. The
1020 data file could even still be in the process of being integrated.
- 1021 • An interpolation method for binary stars that does take into account the
1022 relative velocity of their members.
- 1023 • Indication of size, mass and velocity of the particles by means of the colors
1024 and/or size of their visualization.

1025 Bibliography

- 1026 [1] Opendgl. <http://www.opengl.org>.
- 1027 [2] Vrco. <http://www.vrco.com>.
- 1028 [3] SJ Aarseth. Dynamical evolution of clusters of galaxies i. *MNRAS*,
1029 126(223), 1963.
- 1030 [4] SJ Aarseth. Direct methods for n-body simulations. *Multiple Time Scales*,
1031 page 377, 1985.
- 1032 [5] SJ Aarseth. From nbody1 to nbody6: The growth of an industry. *Publica-*
1033 *tions of the Astronomical Society of the Pacific*, 1999.
- 1034 [6] Reachin Technologies AB. Reachin display. <http://www.reachin.se>.
- 1035 [7] C. Cruz-Neira, Sandin D.J., and DeFanti T.A. Surround-screen projection-
1036 based virtual reality: The design and implementation of the cave. In *SIG-*
1037 *GRAPH 93*, pages 135–142, 1993.
- 1038 [8] J.D. Foley, A. van Dam, S.K. Feiner, and J.F. Hughes. *Computer Graphics*,
1039 pages 478–488. Addison-Wesley Publishing Company, second in c edition,
1040 July 1997.
- 1041 [9] DC Heggie and Mathieu RD. Standardised units and time scales. *Lecture*
1042 *Notes in Physics*, 267, 1986.
- 1043 [10] Jurriaan D. Mulder and Robert van Liere. The personal space station:
1044 Bringing interaction within reach. In *VRIC 2002; 4th Virtual Reality In-*
1045 *ternational Conference*, pages 73–81, Laval, France, June 2002.
- 1046 [11] Ivan R. King. The structure of star clusters iii, some simple dynamical
1047 models. *Astronomical Journal*, 71:64, 02 1966.
- 1048 [12] J Makino. A modified aarseth code for grape and vector processors. *Publ.*
1049 *Astron. Soc. Japan*, 43:859–876, 1991.
- 1050 [13] J.D. Mulder and R. van Liere. Object tracking using projective invariant
1051 marker properties. pages 191 – 198, March 2003.
- 1052 [14] S.F. Portegies Zwart, McMillan S.L., Hut P., and Makino J. Star cluster
1053 ecology - iv. dissection of an open star cluster: photometry. *MNRAS*,
1054 (321):199–226, 2001.

- 1055 [15] C Schmandt. Spatial inputdisplay correspondence in a stereoscopic com-
1056 puter graphic workstation. *Computer Graphics*, 17(3):253–261, July 1983.
- 1057 [16] SGI. The standard template library. <http://www.sgi.com/tech/stl/>.
- 1058 [17] R. Shulakov. <http://staff.science.uva.nl/~rshulako>.
- 1059 [18] T. Poston and L. Serra. Dextrous virtual work. May 1996.
- 1060 [19] T. E. von Wiegand, D. W. Schloerb, and W. L. Sachtler. Virtual work-
1061 bench: Near-field virtual environment system with applications. *Presence*,
1062 8(5):492–519, October 1999.