

Access Time Modeling of a MLR1 Tape Drive

Olav Sandstå, Thomas Maukon Andersen, Roger Midtstraum, and Rune Sætre

Department of Computer and Information Science
Norwegian University of Science and Technology
N-7034 Trondheim, Norway
{olavsa, maukon, roger, runes}@idi.ntnu.no

Abstract

Traditionally, digital tape has been used by applications accessing the tape mostly sequentially. Applications having a random access pattern to the data are much better off storing the data on magnetic disks. Today we see new application areas, where the need for huge amounts of digital storage does not make it cost-effective to store all the data on magnetic disk. One example of such an application is digital video archives. In this paper, we present an access model for a Tandberg MLR1 serpentine tape drive. For such a drive, there is no direct relation between the logical block address and the corresponding physical position on the tape. This makes it difficult to optimize the usage of the tape drive when we have concurrent accesses to the tape. The paper presents a generic model of a serpentine tape drive together with several algorithms for characterizing individual tapes to improve the accuracy of the model. Based on measurements using the MLR1 tape drive the generic model is improved. This extended model makes it possible to estimate seek times with an average error of about two seconds.

1 Introduction

A few decades ago when the blue dinosaurs still ruled the computer rooms, digital tape was used actively in the data processing. Input to the processing (program and data) and the results from the processing were stored on tape. The use of the tape was highly optimized. Most of the processing was done as batch jobs where the tape was read and written sequentially. Since then, magnetic disks have taken over as the primary storage medium for program and data. The main reason is the much better support for random access that a disk can provide. Compared to magnetic disk, digital tape has a random access latency which is 3 to 4 orders of magnitude higher — tens of seconds versus ten milliseconds. As a result, today digital tape is mostly used as a storage medium for backup and distribution, and in applications which need to store really huge amounts of measurement data. One example is NASA's EOSDIS project which will generate more than a terabyte of earth science data per day [7].

The main advantages of digital tape compared to magnetic disks are cost and storage density. For magnetic disks, the cost of the medium is about 1 NOK per MB, while for digital tape, the cost of the medium is about 0.05 NOK per MB. Today, a high-end disk drive stores about 10–20 GB. A digital tape stores the same amount of data on a fraction of the price and storage volume. For new applications like data mining, digital image databases, and video archives, which will require huge amount of data storage, these two factors will be important. Many of these applications will not be able to justify the cost of magnetic disks, making digital tape an alternative.

These applications will have a more random access pattern to the tapes than traditional tape applications have. To minimize the access delay to the tape, and to maximize the utilization of the tape drive, it will be necessary to optimize accesses to the tapes. In this paper we present a model for a digital MLR1 serpentine tape. The purpose of making this model is to be able to optimize concurrent accesses to the tape. The model provides access time estimates which should be used by tape schedulers and other applications accessing the tape. Just as a disk scheduler needs a detailed model of the disk to be able to rearrange the accesses to the disk, a tape scheduler needs an accurate model of the relations between the logical blocks and the physical

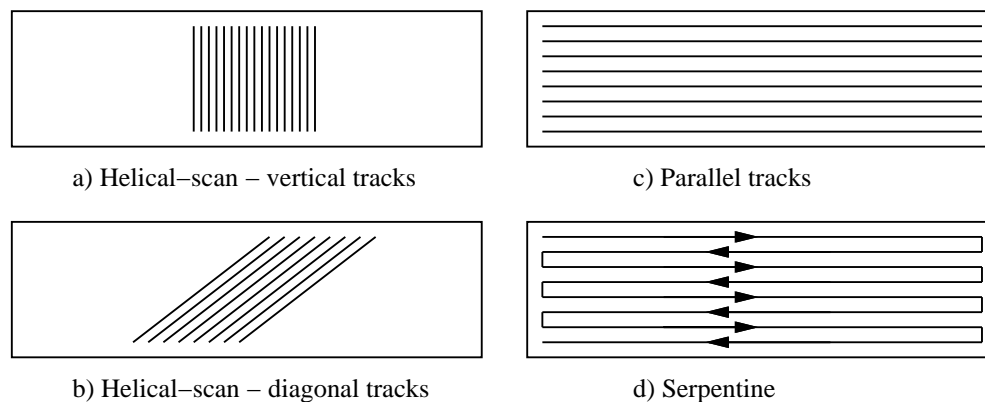


Figure 1 Different tape technologies.

position on the tape to be able to reorder tape accesses. To model a serpentine tape is not trivial, due to the complex format of the tape. There is no direct correspondence between the logical address and the physical position on the tape. By using this model a tape scheduler should be able to reduce the total seek times of concurrent accesses dramatically.

An example can be a digital image database. Assume the database stores high-quality images each requiring 10 MB of storage. A 13GB MLR1 tape will then be able to store 1300 such images. A user doing queries against the database might as a result of a query for images of King Harald find that there exists ten such images on the tape which she wants to access. If the tape is accessed without rearranging the requests, the worst case will occur if the first image is stored on the physical end of the tape, the second image on the physical beginning of the tape, the third on the end, and so on. This will lead to a total of ten full winds of the entire tape. On the MLR1 it takes about 2 minutes to wind the tape from one end to the other, leading to a total time of 20 minutes to retrieve the images. On the other side, if we had an accurate model of the tape and a scheduler using this model, the requests could be rearranged so that all images could be fetched with only one wind/rewind of the tape. It would then take about four minutes to retrieve the ten images, which is eighty percent less time than without a tape scheduler.

The work presented in this paper is done as a part of our work on storage and delivery of digital video [5, 9]. We plan to use digital tape as the main storage for experimenting with digital video archive systems [10].

The rest of the paper is organized as follows. Section 2 gives an introduction to digital tape. Section 3 presents a characterization of the Tandberg MLR1 which is used as a basis for the model. A generic model of MLR1 is given in Section 4. Section 5 presents several algorithms for how the accuracy of the generic tape model can be improved by characterizing the individual tapes. Based on experiments with the generic model, an extended model is presented and evaluated in Section 6. Section 7 contains the conclusions and outline some further work.

1.1 Related Work

A lot of work has been done on modeling and scheduling of accesses on magnetic disks. Not very much has been done on scheduling random accesses on digital tape. Hillyer and Silberschatz [3] presents a detailed model of DLT 4000 tapes. Based on this model they have developed and tested several scheduling algorithms [4]. Using digital tape for storing digital video has been studied by several projects [2, 6, 8], but none of these have studied how to optimize random accesses to the videos stored on the tape.

2 Technologies for Digital Tape

There are two main classes of digital tape technology. The first is *helical-scan*, where the tape drive writes vertical (Figure 1a) or diagonal (Figure 1b) tracks on the tape. The other main technology group is called *linear*, where the data is written in parallel tracks along the direction of the tape [1].

2.1 Helical-scan

Several standards use helical-scan technology. The most well known are 4mm (DAT) and 8mm (video). The popular analog VHS video cassettes also uses helical-scan technology. Tapes using this technology usually obtains high data densities and high transfer rates. The reason for this is that the head is mounted on a rotating cylinder which scans the tape as it passes by. The main problem with this technology is that the rotation of the head imposes severe wear-out on the tape. Because of this wear, the number of times a tape can be read is limited, typically a few hundred to a few thousand passes are guaranteed.

2.2 Linear Tape Technology

There exists two different technologies using parallel tracks on the tape. The classical tapes used by mainframes are called *parallel* because all the tracks along the tape are written and read in parallel (Figure 1c). In this paper we focus on the *serpentine* model. A serpentine tape drive writes one track (or group of tracks) down the tape (forward direction), then writes the next track (or group of tracks) in the opposite direction (reverse direction) as seen in Figure 1d.

Today we have two main technologies for serpentine drives, DLT and QIC. DLT – Digital Linear Tape – is a technology developed by DEC (Digital Equipment Corporation). It uses a half inch tape which is stored in a cassette having only one reel. The second reel is a part of the tape drive. When inserting a DLT tape into a drive the tape first has to be mounted onto this reel. The current top model of the DLT series is the Quantum DLT 7000. This tape drive stores 35 GB (uncompressed) data and has a transfer rate of 5MB/s.

QIC – Quarter Inch Cassette – started as a standard for making inexpensive tape storage with modest capacity and bandwidth, but during the last years the storage density has been increased and QIC are now approaching DLT in both storage capacity and data transfer bandwidth. As the name implies the width of the tape is an quarter inch, i.e., only half the width of the DLT tape. Opposite of the DLT, QIC tapes has both reels inside the cassette. QIC provides standard tape storage formats covering the range from 60MB to 13GB.

2.3 Tandberg MLR1

In the experiments described in this paper we have used the Tandberg MLR1¹ drive. This drive is based on the new 13GB QIC standard. In this subsection we will give a brief introduction to this drive. More details can be found in [11].

A MLR1 tape is able to store 13 GB of uncompressed data. The tape drive can deliver (read/write) a maximum sustained data rate of 1.5 MB/s to/from the host. The tape technology used is serpentine. Each tape consists of 144 data tracks. The tracks are grouped in pairs, i.e., two tracks are written in parallel in each scan of the tape. This gives us 72 logical tracks, 36 in forward direction and 36 in reverse direction.

Data is stored in frames on the tape medium. Each frame consists of 52 data blocks and 12 Error Correction Characters (ECC) blocks. Each block stores 512 bytes of data. The drive supports the SCSI interface. Applications using the drive, read and write logical blocks of data to the drive. These logical blocks can be fixed or variable length blocks. There is no direct mapping between the logical block address and the physical position on the tape. The reason for this is that during writing of a tape there will be bad spots on the tape. The drive automatically skips such areas. As we will see in the next sections this influences on the modeling of the seek times for a given tape.

3 Characterization of a MLR1 Tape

The purpose of this paper is to establish a model of the access times for a Tandberg MLR1 tape. This model should be able to answer questions like “*The tape is now positioned at the start of the tape, how much time does it take until we are at the start position of logical block X*” or “*How much time will it take to seek from logical position X to logical position Y*”. In [3] this function is called `locate_time(src, dst)`.

¹MLR is short for Multi-channel Linear Recording.

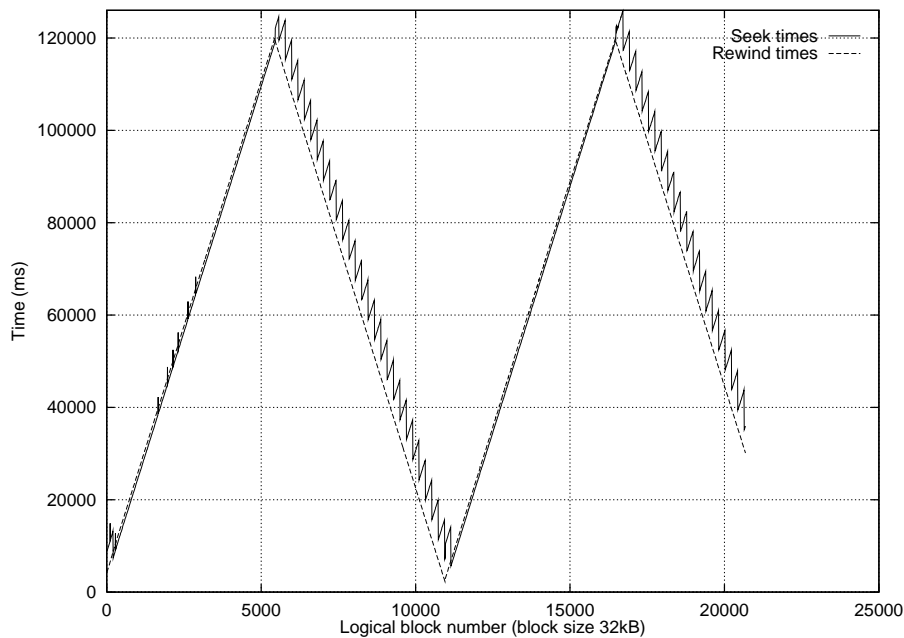


Figure 2 Seek and rewind times for the first tracks of the tape.

To determine this model we have run several experiments using the MLR1. Before we started the experiments we filled the tapes with fixed length logical data blocks. Each logical block was 32kB. The number of blocks we were able to write to a tape varied from about 398000 to about 400100 blocks. The average write time for a block was 22 milliseconds. To write a full tape takes approximately 2.5 hours.

Figure 2 shows the seek and rewind times for the three and half first tracks. Along the x-axis is the logical block number, and the y-axis contains the number of milliseconds it takes to seek from the start of the tape to that logical block. These numbers are found by rewinding the tape to the start position, then measure the time it takes the tape drive to seek to the specified block. We also measured the time to rewind the tape back to the start position. We did this for every eight blocks along the x-axis².

From the figure we see that 1) for forward tracks the curves for seek and rewind times both are straight and overlaps, 2) for reverse tracks the curve for seek times have a sawtooth pattern, while the curve for rewind times are straight.

To explain the sawtooth pattern we have enlarged a small section of the reverse track in Figure 3. This figure shows that each 'tooth' consists of a straight line representing about 200 logical blocks. The reason we get this pattern on the reverse tracks is that these tracks have to be read in the opposite direction of the forward tracks. When the tape drive tries to locate a position on a reverse track starting from the start position it first has to seek past the sought block, and then start reading in the read direction until it has found the sought block. Figure 3 indicates that the tape drive uses a set of predetermined points to decide where to stop the seek in forward direction and start seeking in the opposite direction. These points corresponds to the first block in each sawtooth.

Hillyer and Silberschatz [3] experienced similar sawtooth patterns for the DLT 4000 drive. They defined the points where the seek time has a big dip from one sawtooth to the next as the *key points* of the tape. But opposite what we found, they also experienced sawtooth patterns along the forward track. The reason for this is that the DLT 4000 uses one speed (seek speed) for locating the correct key point block, and a slower speed (read speed) for locating the correct block between two key points. Tandberg MLR1 uses the same speed for both seeking and reading. This suggests there will be key points along the forward tracks too, and by doing seek operations which starts on the physical end of the tape (instead of the beginning) and plotting the seek times we see sawtooths on the forward track too.

To get statistical data about the seek times we ran two experiments. In the first experiment we measured the seek times from the start of the tape to a random position on the tape. In the second experiment we

²Still the tape drive ran continuously for about 2 days to make this graph.

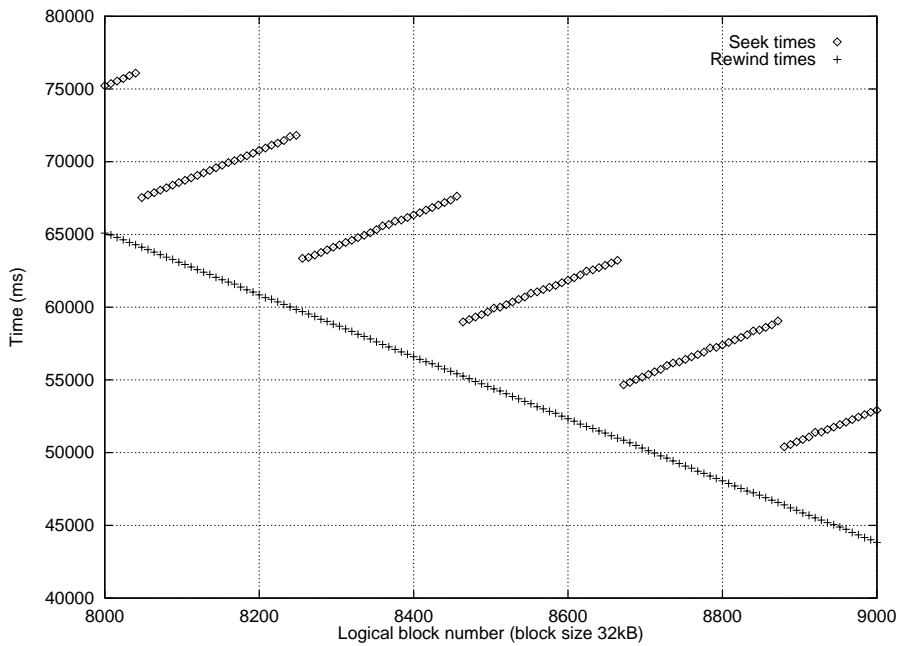


Figure 3 A small section of the first reverse track.

	Seek from start position	Seek from random position
Minimum	4.02 seconds	0.85 seconds
Average	65.4 seconds	45.5 seconds
Maximum	126 seconds	125 seconds

Table 1 Seek times for seeking from the start of the tape to a random position and from a random position to another random position on the tape. These numbers are based on about 8000 seeks on the tape.

measured the seek time from a random position on the tape to another random position on the tape. The results are given in Table 1. These numbers together with the experienced curves presented in Figure 2 will be the basic for making a generic model for the seek times.

4 A Generic Model for Seek Times

To be able to estimate the seek times of the MLR1, we need a model which describes the behavior of the tape drive. As a first approximation to such a model we will present a generic model of a serpentine tape drive. This model is based on the observations presented in the previous section.

The model is based on the assumptions that the tape is already mounted in the tape drive and that the

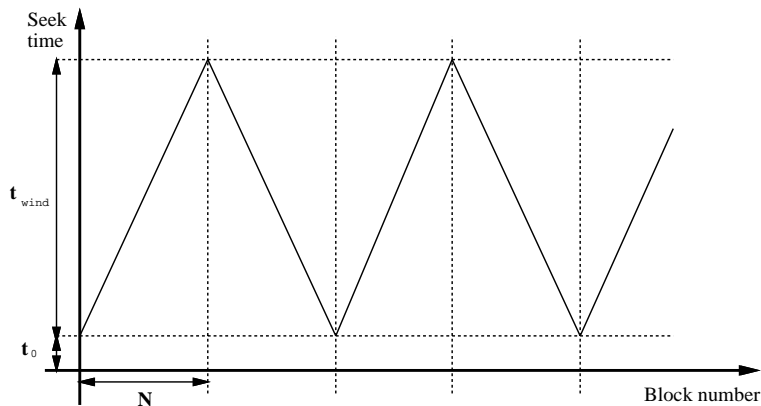


Figure 4 A generic model for characterization of a serpentine tape.

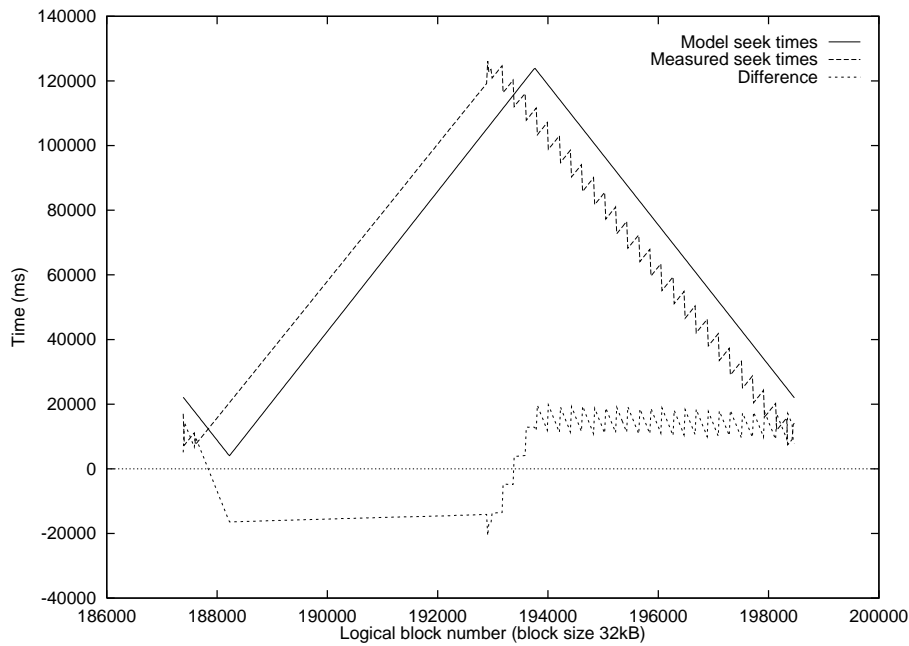


Figure 5 The measured and estimated seek times for two tracks using the generic tape model.

tape consists of fixed sized blocks. Further, we assume that each track of the tape consists of a fixed number of blocks. This is not entirely correct, because the number of blocks along a track might vary due to bad areas on the tape. We will discuss this further in the evaluation of the model.

This simplified, generic model is shown in Figure 4. The model gives the seek times from the start of the tape to any given position. The parameters which instruments the model are:

- N is the number of blocks in each track.
- t_0 – the measurements of seek times show that even the shortest seeks take a minimum time to perform.
- t_{wind} is the time the tape drive needs to wind the tape from the start of the tape to the end of the tape.

Based on the model in Figure 4 we can find an expression for the *physical* position a given *logical* block has on the tape:

$$pos = | [(blocknumber + N) \bmod 2N] - N |$$

By dividing this equation by N we get the position as a number between 0 and 1. By using this we can express the seek time between two random positions on the tape as:

$$t = t_0 + \frac{t_{wind}}{N} (pos_{stop} - pos_{start})$$

4.1 Validation of the Model

To use this model we first have to estimate the values of t_0 , t_{wind} , and N . As an approximation for N we take the number of blocks written to a tape divided by the number of tracks on the tape. Unfortunately the number of blocks per tape vary rather much. For the tapes we have used the number of blocks written has been between 398082 and 400055 blocks per tape. This gives an N in the interval from 5529 to 5556. The average number of blocks per tape has been 398637, giving an average value for $N = 5537$.

From the previous section we use the minimum seek time as $t_0 = 4.0$ seconds. In theory, it should take 120 seconds to wind the tape from one end to the other³. Measurements show that this is a very good

³The length of the tape is 1200 feet and the speed of the tape is 120ips (inches per second).

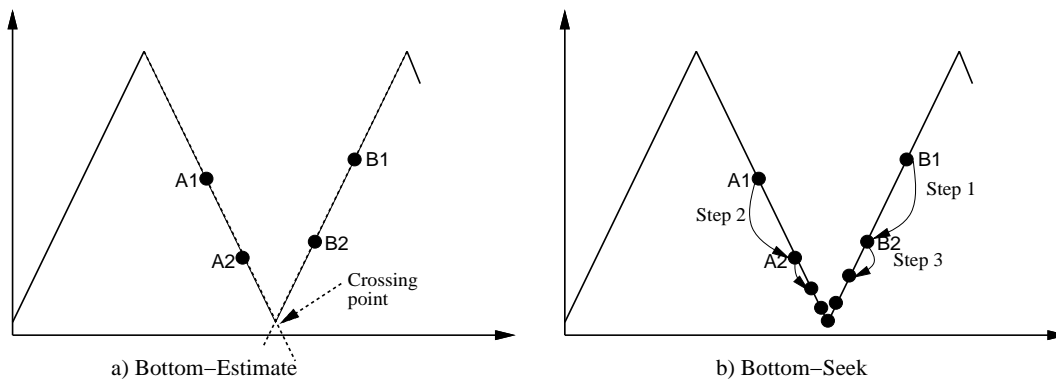


Figure 6 An overview of the Bottom-Estimate and Bottom-Seek algorithms.

approximation. We have measured the maximum rewind times to be between 119.9 and 121.2 seconds. Based on this, we select $t_{wind} = 120.0$ seconds.

To validate this model we picked 1000 random block numbers and measured the seek times. We compared the measured seek times to the corresponding seek times suggested by the generic model (see Figure 4), and computed the errors between them. The average error between the measured and estimated seek times we found to be 12.5 seconds.

In Figure 5 we have plotted the measured and estimated seek times for two tracks on the tape, together with the error difference. This figure shows that the the estimated curve do not model the curve for the measured seek times very well. The reason for this is that we use a fixed track length N in the model. On real tapes the track length will vary from track to track and from tape to tape. This will make the curve for the estimated seek times *drift* away from curve of measured seek times. As a result the difference between measured and estimated seek times will increase as a function of the logical position on the tape.

5 Modeling Individual Tapes

The generic model got worse as we got farther out on the tape. The reason was that we did not know the exact number of blocks per track. To make a model without the drifting problem we need to characterize each individual tape. The straight forward way to characterize a tape completely, would be to perform a seek from the start of the tape to each of the blocks on the tape. Unfortunately, this is not feasible, since it would take more than a year to perform this for a single tape. One way to improve the accuracy of the generic model will be to find better estimates for the number of blocks per track on each tape. In this section, we will present and compare several strategies for how this can be done.

A first approximation of the number of blocks per track can be found by dividing the exact number of blocks written to the tape by the number of tracks on the tape. This can only be done if the entire tape is filled up by fixed sized blocks. We will call this strategy *Exact tape-length*.

To further improve the generic model, we try to identify the addresses of the blocks on the end of each track. These addresses will vary from tape to tape, due to varying number of bad blocks per track. Based on these addresses, we can estimate a correct value for each track length. We have tested two different strategies for estimating the track lengths. The first strategy is based on using the write times of the tape, while the second strategy finds the end of the tracks by performing operations on the tape.

If we have control of the writing of the tape, and the tape is written block by block, we can measure the writing time for each block. Writing a 32kB block to the tape takes in average 22ms, but every time the tape reaches the end of a track, it has to stop the tape motion before it can start writing in the opposite direction. By studying the writing times, we find that this change of direction takes about two seconds. This can be used to locate the address of the end points of the tracks rather accurate. We call this strategy *Write-Turn*.

If the tape is already written by somebody else or by an application which do not let us have access to the write times, we have to locate the end of the tracks by performing operations on the tape. One way to do this is to estimate a block interval which cover each end of a track, and by doing measurements on this interval gradually decrease the size of this interval until we have found the last block in the track.

Strategy	Average error [<i>seconds</i>]				Standard deviation		
	All	Tape 1	Tape 2	Tape 3	Tape 1	Tape 2	Tape 3
Generic model	12.5 s	16.2 s	10.2 s	11.0 s	10.5	8.59	5.11
Exact tape-length	4.71 s	3.35 s	3.83 s	6.97 s	2.90	2.45	4.71
Write-turn	3.37 s	3.42 s	3.43 s	3.25 s	2.86	2.96	2.69
Bottom-estimate	3.61 s	3.68 s	3.63 s	3.51 s	2.88	2.97	2.73
Bottom-seek	3.79 s	3.84 s	3.85 s	3.67 s	2.90	2.98	2.74

Table 2 Results from testing the algorithms on three tapes. The table contains the average difference between measured seek times and seek times estimated by the tape model.

Since each MLR1 tape contains 72 logical tracks we have to perform this process for each of these. To reduce the total time for characterizing each tape we optimize this further to only find the end of the 36 tracks which are closest to the physical start of the tape. This will reduce the time to find the tracks significantly because a seek/rewind measurements for a block close to the start of the tape takes from 8 seconds and upwards, while seeking for blocks at the end of the tape takes about 250 seconds.

We have implemented and tested two algorithms for estimating the last block on each of the reverse tracks. Both starts by estimating an end block for this track. This estimate is based on adding two times the average track length to the end point of the previous reverse track. Both algorithms are based on the property that the sought block should be the block with the smallest seek time, i.e., blocks surrounding it should have higher seek times.

Bottom-estimate. Based on the estimated end point we select two points on each side of the end point (see points A1, A2, B2, and B1 in Figure 6a). These points must be selected sufficiently away from the estimated end point to be sure that the real end point is between them. We then perform a seek to each of these positions to measure the seek and rewind times for each of these block addresses and compute the crossing point between the lines through each of them. This crossing point is used as an estimate for the last block on the track. To make this algorithm more sturdy against a bad rewind time measurement we compute the slope of the lines through the points. If this slope is not within a certain interval we do a new estimate of that line based on two new measurements done on two slightly different tape positions.

Bottom-seek. This is based on a variation of binary seek. We select one point on each side of the guessed end block, A1 and B1 (see Figure 6b) and measure the seek/rewind times. The one with the highest rewind time is replaced by a block lying inbetween these two points (binary seek step). This process is repeated until we have reduced the length of the interval between the two blocks below a predetermined threshold (e.g., 8 blocks). Then the block in the middle of this interval is used as the estimate for the last block on the track.

An important point is that when we compare relative position of the blocks on the tape, we use the rewind times instead of the seek times. The reason for this is that the rewind times will be on a straight line for each track, and this makes it easier to compare them. The seek times for reverse tracks are much more difficult to compare due to the sawtooth shape of the curve (see Figure 2).

5.1 Validation of the Algorithms

To be able to compare these algorithms with the generic model presented in the previous section we filled three tapes with 32kB blocks and logged the write times for each block. From the writing of the tapes we got the exact number of blocks on each tape and found the end address of each track. On each of these tapes we used the the two algorithms described above to find the end blocks of the reverse tracks on the tape. These block numbers were then used for estimating the track length for each track on the tape. We used these track lengths to adjust the generic model. Instead of having one fixed N value for the model, we now used the track lengths found by the algorithms for each individual track on the tape.

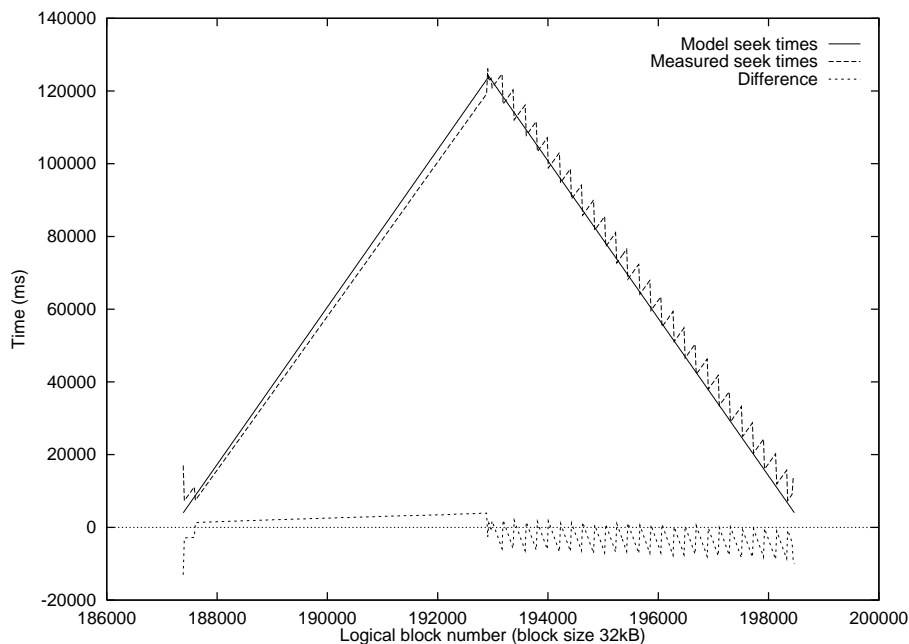


Figure 7 Measured and estimated seek times for two tracks using the adjusted track lengths found by the *Write-Turn* algorithm.

As input to the adjusted model we used the same 1000 random blocks as used for validating the generic model, and compared the results with the measured seek times. The results for the four strategies are presented in Table 2 together with the results for the generic model. Compared to the generic model we see that all strategies perform better. In our experiments we find that the average error between the measured seek times and the estimated seek times are less than 5 seconds for all strategies compared to 12.5 seconds for the generic model. We also see that the three last strategies which estimates the length of the individual tracks performs better than the strategy where we use a constant track length based on the total length of the tape. All of the three strategies have an average error between 3 and 4 seconds. In Figure 7 we have plotted the measured and estimated curves for the same two tracks as shown in Figure 5 using the *Write-Turn* strategy. This time we observe that the two curves overlap much better leading to much better estimates. We have also plotted the curve for the error of the model.

If we compare the three strategies for detecting the ends of the tracks, we see that *Write-Turn* gives the best results. The reason is that *Write-Turn* uses the exact end points detected during writing, while the two other only estimates the bottom to be within a certain interval. Comparing *Bottom-Estimate* and *Bottom-Seek*, we see that they perform almost identically, with *Bottom-Estimate* performing slightly better.

It is important to be aware of the cost of getting these better models. We now have to model each individual tape separately. The *Exact tape-length* strategy do only require that we get the total length of the tape when it is written. The *Write-Turn* strategy requires that we are able to measure the writing times of the blocks on the tape. The two last algorithms require that each tape are run through the process of finding the end of the tracks before the model can be used. This is rather time-consuming. These required time usage are respectively 75 and 131 minutes for the *Bottom-estimate* and *Bottom-seek* algorithms.

6 Extended Model

In the previous section, we presented several algorithms to improve the generic model to get better estimates by characterizing each individual tape. In this section, we will present how we have used the results from the evaluations of the algorithms in the previous section, to improve the *generic model* itself. We will call the new model an *extended generic model for MLR1*.

From studying the results in the previous section (see Figure 7) we did the following observations:

1. For forward tracks the model consistently overestimates the seek times.

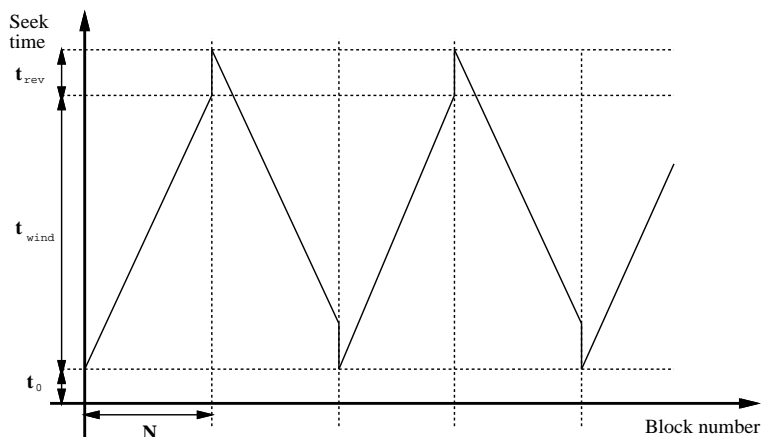


Figure 8 An extended model for characterization of a MLR1 tape.

Strategy	Average error [<i>seconds</i>]				Standard deviation		
	All	Tape 1	Tape 2	Tape 3	Tape 1	Tape 2	Tape 3
Extended model	12.0 s	13.9 s	8.75 s	13.5 s	10.0	7.50	5.34
Exact tape-length	5.35 s	2.29 s	5.20 s	8.55 s	1.79	3.01	5.57
Write-turn	2.30 s	2.32 s	2.47 s	2.12 s	1.91	2.12	1.89
Bottom-estimate	2.25 s	2.26 s	2.40 s	2.07 s	2.06	2.26	1.98
Bottom-seek	2.23 s	2.26 s	2.37 s	2.06 s	2.15	2.41	2.02

Table 3 Results from testing of the algorithms on the extended model of a MLR1 tape. The table contains the average difference between measured seek times and seek times estimated by the extended model using different algorithms for instrumenting the model.

2. For reverse tracks the model consistently underestimates the seek times.

Based on these observations we did the following extensions to the generic model:

- We have reduced the value of t_0 from 4 to 2 seconds. The reason for using 4 seconds was that this was the least seek time we have experienced using the tape drive. But as can be seen in Figure 2 the first part of the curve for seek times on forward tracks has some extra delay in the seek times compared to the rest of the curve for seek times. The reason is that the tape drive always does some extra tape moving for the first blocks on each forward track. To avoid that the model overestimates seek times for forward tracks we have reduced t_0 .
- We have increased the seek times for all positions on the reverse track by a certain amount which we call t_{rev} . This is for compensating for the extra time to find a position on the reverse track. First we always have to change the winding direction once for each seek. By studying the seek times we have found that the time to change winding direction takes approximately two seconds. In addition the tape drive will always wind to a key point before it changes direction. This means that for each seek for a position on a reverse track the tape drive seeks longer than actually needed. On average this will be half of the “high” of a sawtooth. The “high” of a sawtooth is about 4 seconds (see Figure 3). This extra distance we go twice, first in forward direction until we have found the key point, then in reverse direction. Totally this gives t_{rev} to be 6 seconds.

This extended model is shown in Figure 8.

6.1 Evaluation of the Extended Model

To evaluate the extended model we ran the same tests as we did for the algorithms presented in the previous section. The corresponding results can be found in Table 3.

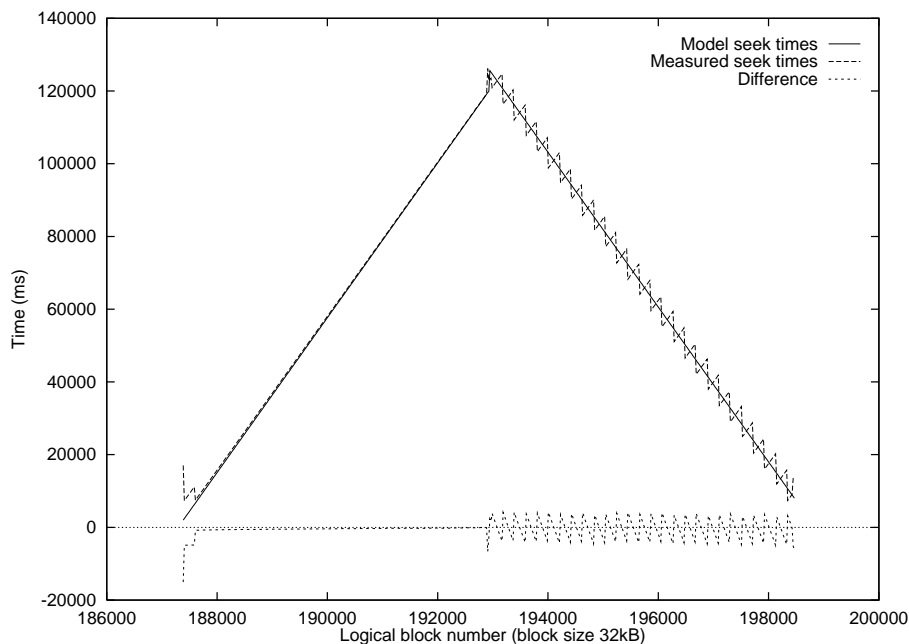


Figure 9 Measured and estimated seek times for two tracks using the adjusted track lengths found by the Write-Turn algorithm for the extended tape model.

We see that we get similar results as we did when we used the algorithms on the generic model. For the three algorithms which locates the individual tracks on the tape we find that all performs almost identical. The average error for these algorithms is about 2.3 seconds, which is 30 percent better than the generic model. A plot of the measured and estimated seek times for two tracks on the tape can be seen in Figure 9.

For the algorithm which uses the exact tape length, we see that the extended model actually performs worse in some cases. This happens when the model has drifted to the right of the seek curve (as seen in Figure 5).

7 Conclusion

In this paper we have investigated the seek times of a Tandberg MLR1 tape drive. The goal of the work has been to establish a model for estimating seek times for random accesses to the tape. To model the seek times for the MLR1 tape is non-trivial because there is no direct relation between the logical block addresses used by applications and the corresponding physical position on the tape. By studying the seek and rewind times we have proposed a generic model of a MLR1 tape. Unfortunately this model does not give very accurate results because it is based on fixed track lengths. The reason the generic model performs badly is that the length of the tracks vary from track to track and even more from tape to tape.

To be able to give more exact estimates for seek times it is necessary to characterize each individual tape. We have evaluated four strategies for characterizing tapes. Each of these strategies have different qualities and costs. Using the *Exact tape-length* is the easiest and least costly, but gives the least accurate estimates. It also requires that the tape is filled completely.

The other strategies locates the end positions of each track on the tape. They all give approximately the same accuracy for the estimated seek times, but they have very different costs for establishing the model. All of them requires that we for each tape keep a list of the addresses of the end tracks. If we know the writing times of each block written to the tape, the *Write-Turn* strategy gives the best model for estimating seek times. It is also the least costly of the three strategies. If we do not have access to the write times, the *Bottom-Estimate* strategy is the preferred method for instrumenting the tape model. Unfortunately the cost of establishing the list of the end positions of each track requires about one hour of measurement of seek/rewind times for each tape. The *Bottom-Seek* algorithm performs almost identically to *Bottom-Estimate*, but the cost of using the algorithm to find the list of end positions is higher. By using one of these strategies we are able

to estimate seek times with an average error of two to three seconds.

To further improve the model we have to include the sawtooths into the model. To locate each of the sawtooths on the tape is not feasible because it would require too much time to characterize each tape. An alternative could be to extend the tape drive with commands for letting users read information about the key points from the drive.

The model presented in this paper gives estimates for seek times from the beginning of the tape to a given position on the tape. We plan to further improve the model of the tape and use this as a basis for experimenting with scheduling of concurrent accesses to the tape. To be able to do this we have to extend the model to make it able to estimate seek times between two given positions on the tape. The tape model together with a scheduler for tape requests will be used as a part of our ongoing work on storing digital video on digital tape for video archive applications.

Acknowledgment

We would like to thank Karel Babicky and Bernt Breivik, both Tandberg Data, for commenting on technical questions we have had about the usage of the MLR1.

References

- [1] K. Bratbergsengen. *Filsystemer. Lagring og behandling av store datamengder*. Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway, 1997. In Norwegian.
- [2] A. L. Chervenak, D. A. Patterson, and R. H. Katz. Choosing the best storage system for video service. In *Proceedings of ACM Multimedia '95*, pages 109–119, San Francisco, California, November 1995.
- [3] B. K. Hillyer and A. Silberschatz. On the modeling and performance characteristics of a serpentine tape drive. In *Proceedings of the 1996 ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, pages 170–179, Philadelphia, Pennsylvania, May 1996.
- [4] B. K. Hillyer and A. Silberschatz. Random I/O scheduling in online tertiary storage. In *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 195–204, Montreal, Canada, June 1996.
- [5] R. Hjelsvold, S. Langørgen, R. Midtstraum, and O. Sandstå. Integrated Video Archive Tools. In *Proceedings of ACM Multimedia '95*, pages 283–293, San Francisco, California, November 1995.
- [6] K. Keeton, A. Drapeau, D. Patterson, and R. H. Katz. Storage alternatives for video service. In *IEEE 13th Symposium On Mass Storage Systems*, pages 100–105, Annecy, France, June 1994.
- [7] B. Kobler, J. Berbert, P. Caulk, and P. Hariharan. Architecture and design of storage and data management for the NASA Earth Observing System Data and Information System (EOSDIS). In *IEEE 14th Symposium on Mass Storage Systems*, pages 65–76, Monterey, California, September 1995.
- [8] S. Lau, J. C. Lui, and P. Wong. A cost-effective near-line storage server for multimedia system. In *Proceedings of the Eleventh International Conference on Data Engineering*, pages 449–456, Taipei, Taiwan, March 1995.
- [9] O. Sandstå, S. Langørgen, and R. Midtstraum. Video server on an ATM connected cluster of workstations. In *Proceedings of XVII International Conference of the Chilean Computer Science Society*, November 1997.
- [10] G. Sørensen. Server for digital video archives. Master's thesis, Department of Computer and Information Science, Norwegian University of Science and Technology, Trondheim, Norway, March 1997. In Norwegian.
- [11] Tandberg Data, Oslo, Norway. *Tandberg MLR1 Series Streaming Tape Cartridge Drives Reference Manual*, 1 edition, August 1996.