MASTER

Design of an automated system for geometry adjustment of television sets and monitors

van Mierlo, G.A.

Award date:
1988

Link to publication
DESIGN OF AN AUTOMATED SYSTEM FOR GEOMETRY ADJUSTMENT OF TELEVISION SETS AND MONITORS.

A report of graduating work at the Philips Centre For manufacturing Technology, performed by G.A. van Mierlo (id.nr.195978), coached by Ir. D. Bruyland.

SUMMARY

To graduate at the Electrical Engineering Department of the Eindhoven University of Technology I worked on an automation project at the Centre For manufacturing Technology of Philips Eindhoven. An automated system for adjustment of the geometry of monitors and t.v. sets had to be designed. This adjustment has been done manually until now; inspection was done by human eyes. The Automated Geometry Adjustment System (AGAS) had to fit in the AMTS concept and amongst others had to use MOBUS instrumentation. Other limiting factors were maximum capital costs and time allowed for the adjustments. A total set of requirements was gathered from all kind of sources like conventional adjustment prescriptions and product managers.

The total AGAS was split up into sub systems, each representing a separate main functionality. For each of the sub systems alternatives were investigated. From these alternatives a system was composed based on an IBM personal computer or compatible computer. The configuration contains instrumentation from two standardized computer controlled instrumentation families that have been developed at the CFT, MOBUS and PAPS. The inspection of the picture on the t.v. or monitor (also called Device Under Test or DUT) is realised using a CCD camera.

All errors present in the AGAS were analyzed to be sure that it would function accurately enough.

Both hardware and software were set up first to obtain a system that could be used in experiments. Setting up software meant that AMTS software, which had been written for another computer and operating system, was converted to a version applicable to the IBM PC and some additional software was created. When this had been done control algorithms were designed. As far as it can be seen all constraints are met in the AGAS. Although the project hasn't been finished yet; it will be continued by others. The control algorithms have to be tested and the vision problem needs to be solved. Recommendations are given which resulted from the error analysis and built up experience.

A good start has been made to develop an automated geometry adjustment system, although several things still have to be accomplished before it will be a product that is commercially viable.
VII CONTROL ALGORITHMS

VII.1 Starting points

VII.2 The deflection circuit in a 3A t.v. set

VII.3 Algorithms for geometry adjustment of 3A sets
  VII.3.1 Order of alignment
  VII.3.2 Specification of the algorithms
  VII.3.3 Time and accuracy consideration

VII.4 Conclusions

VIII RECOMMENDATIONS FOR FOLLOW-UP

IX CONCLUSIONS

BIBLIOGRAPHY

APPENDIX A: POWER SUPPLY REQUIREMENTS

APPENDIX B: FUNCTIONS INTERFACED TO μAMTS

APPENDIX C: DEFAULT SETTINGS
I. INTRODUCTION

In modern production lines the assembly of products is realised in a short cyclic process. Amongst others this concept is employed in the production of monitors and t.v. sets at Philips. In order to obtain well performing products from such line, things like focus, colour balance and geometry have to be adjusted after assembly. At the moment these adjustments are done manually, with or without use of measuring tools. This, however, always leads to a subjective final judgement. For obtaining objective adjustments, and with a view to desired automation of the production process at the same time, at the Philips Centre For manufacturing Technology (CFT) a system has to be designed with which it is possible to adjust the quantities just mentioned automatically.

The subject of this graduating work is the design of a tool with a part of this total functionality. The adjustment of the geometry is dealt with in this report. The system designed will be called AGAS (Automated Geometry Adjustment System) from now on.

Several conditions have to be met in the AGAS. It has to be flexible to cover a variety of products to be adjusted (varying screen sizes, electronics). The maximum time allowed for the geometry adjustments is 25 seconds. Re-programming has to be very easy, preferably using a menu structure. The AGAS has to fit in a software concept called AMTS and a hardware concept named MOBUS. These concepts have been accepted as standards inside CFT. The total capital costs mustn't exceed Hfl 50,000.-- .

In order to make the situation less complex and time consuming, thus being more acceptable as a graduating activity, a restriction is laid on the product to be handled. Initially only t.v. sets typed 3A are considered to be adjusted. This choice is made because these sets are produced on large scale, its production line thus being considered for automation very quickly. An advantage of these sets is the fact that they are fully microprocessor controlled internally by means of the so called IIC protocol. This will simplify stimulation of the set, because a MOBUS instrument card for controlling the IIC bus has already been developed.

First of all some basic concepts applied at the Philips CFT which will be relevant are introduced. Then system requirements are formulated. The AGAS is split up into functional sub systems to simplify the problem of configuring it. Alternatives are investigated for each of the sub systems. A definite system is composed from these alternatives.

Errors are analyzed to see whether the chosen system will be functioning accurately enough to fulfill the requirements. Some activities are needed to make the system usable. The outline of the AGAS, including all actions taken to realize it, is discussed.

An algorithm is designed to adjust the geometry of the t.v. set using the available means. Not much attention is paid to the vision problem. Considerations concerning the picture acquisition are mentioned, though, which may be very useful in the continuation of the project.
At Philips, partly inside the Centre for manufacturing Technology, several standard concepts have been defined and implemented, concerning both software and hardware. Some of these concepts are relevant or will become relevant to the development of an automated system for adjusting geometry of a TV set. In this chapter four concepts will be introduced: AMTS (software), MOBUS (instrumentation), IIC (computer controlled devices) and PAPS (vision). The first three concepts were already mentioned in the introduction. They are part of the boundary conditions for the AGAS. The last has not been mentioned yet, but will be relevant too in this application.

### II.1 AMTS

One of the requirements to be fulfilled, is that the AGAS has to be implemented in a software structure called AMTS. AMTS stands for Architecture for Modular Test Systems. It is a system-software package designed to support the development, maintenance and running of all software in signal processing activities, mainly in industrial production environments. It supplies the necessary tools to achieve cost effectiveness in Automatic Testing.

In automatic testing there are a Device Under Test (DUT), Automated Test Equipment (ATE), and experts for each of these. AMTS creates a situation where the DUT expert can concentrate on testing matters without worrying about software or computer problems; these can be solved by the ATE expert, independently. AMTS provides ATEs with DUT expert-oriented functions, which frees the DUT expert from low-level control of instruments. A structured programming language is used and user-friendly facilities for interactive programming and debugging are available. The ATE expert, with AMTS system knowledge, can construct drivers for a range of instruments in an easy way and he can configure ATE with required functionality.

AMTS is part of a layered structure in the software of automated test equipment. The integration of AMTS in the layer structure of an ATE is pointed out in the following picture.
The core of the structure is formed by the ATE hardware (computer and instrumentation). In the design of AMTS the hardware was thought to be based on an 68000 microprocessor system, but it can also be configured around an IBM-PC compatible machine. For the IBM-PC a special version of AMTS is being created at the moment, which is called \( \mu \text{AMTS} \). This development will be completed in the near future.

On top of the computer hardware resides the operating system. For the 68000 the operating system used is called DRM (Distributed Real-time Monitor), which has been developed by Philips. It is a modern operating system (real-time, multi-tasking and multi-processing). On the IBM-PC compatible machine the operating system is called DOS. This operating system is single tasking and single processing. In the near future an operating system called OS/2 will be available. This operating system will have multi-tasking capabilities.

The next layer is AMTS which implies the Interactive Programming Environment. This is an environment in which the user can write programs, debug them easily and interpret them; it contains an editor, tracer, debugger, interpreter, file manager, etc. The programming language is called TPL (Test Programming Language). It has been based on British Standard 6192 Pascal and is extended with all kind of extra language features. Amongst others it contains features that have been implemented specifically for signal processing (e.g. extensive bit manipulation and commands for controlling instrumentation). The use of modules is admitted; one can add or subtract modules to or from the programming buffer. These modules can be used (called) by one main program, which will also be a module in the buffer. All modules have to be written in TPL.
The IPE has extensive help facilities and a syntax checker in the editor. A compiler has not been implemented yet, but will be developed in the future.

AMTS is a SW package written in C- and Assembly language. The AMTS layer is built by compiling and linking a lot of sources, all contributing to the functionality of AMTS. The instrument drivers made by the ATE expert are part of it, thus supplying specific commands to the DUT expert for the use in test programs to control the test equipment. Hereby the AMTS layer also interfaces directly with the hardware.

On top of all this the application is situated. This is the TPL program that is solving the test problem.

II.2 MOBUS

MOBUS stands for Modular Building System. It is a modular system designed to build test and measurement installations out of a range of functional modules. The standard format for these modules equals the dimensions of a double Euro card. The modules can be divided into a number of categories, the most important of these being:

- system supporting modules (like a CPU card)
- measuring modules
- stimuli modules
- switching modules
- general analog input/output modules
- general digital input/output modules

There are two buses on the back panel of a MOBUS rack, named the General Purpose Bus Simplified (GPBS) and the System Control and Supply bus (SCS bus). The GPBS and the SCS bus take care of all system connections to a card, whereas application connections are made via connectors on the other side of the card.

The GPBS contains a 16 bits data bus, an 8 bits address bus, some timing signals, a reset line, interrupt lines, break lines, a READ/WRITE line, a WORD/CHAR line, a "function accepted" line, a +5V, -12V and +12V supply. The SCS bus contains test lines for a self test of the system, interrupt lines, emergency lines to check whether all cards are present and have their supply voltage, and some additional control signals which will not be mentioned here explicitly. It also contains a range of supply voltages (+5V, +15V, -15V, +24V and -24V).

In fact, the names of the two buses are not right because the GPBS contains connections for control and supply, as well as the SCS bus. This situation is caused by the fact that the GPBS has been derived from the General Purpose Bus (GPB), that had already been defined for P800 computer systems (supplied by Philips), when the development of the MOBUS was started. Initially, MOBUS was designed for use with P800 series computers. For these computers a special software package has been developed; a system controlled by a P800 computer can be programmed using the ACTS language (Architecture of Computerised Test Systems).

Nowadays, it is possible to use other computers, e.g. an 68000 or an IBM-PC; one can interface between the computer hardware and the GPBS by applying the right bus translation card. When using one of these computers, AMTS is used as the programming environ-
ment for the test system. Instrument drivers for all existing MOBUS cards have been written for 68000 systems. Examples of MOBUS cards are a programmable oscillator card, a video timing generator generating video signals which can be programmed, cards containing matrices of (coaxial) switches, motor control cards, a card for interfacing to the IIC bus, a general purpose measuring card for measuring analog signals, a digital input card and a signal processing card (e.g. for DFT). The MOBUS product range also includes appropriate mounting hardware and power supplies.

II.3 IIC

IIC is an abbreviation of Inter Integrated Circuit. It is a communication concept designed for applications in microcomputer controlled systems where the costs of connections have to be kept to a minimum and a low-speed data transfer is admitted. It has been developed in an attempt to standardise the design of low-cost equipment, mainly consumer applications. Typical application areas are car radio and t.v. sets.

For all IIC communication the IIC bus is used. The IIC bus consists of three wires: a serial data wire (SDA), serial clock wire (SCL) and a ground wire. A lot of devices, e.g. all kinds of special purpose processors and memories, are connected to these two wires in a system. Every device has its unique address - whether it is a microcomputer, LCD driver, memory or keyboard interface - and can operate as either a transmitter or receiver, depending on the device considered. Obviously, a LCD driver is only a receiver, while a memory can both receive and transmit data. In addition, devices can also be considered as masters or slaves when performing data transfers. A master is the device which initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time any device addressed is considered a slave.

As an example, the block diagram of an IIC controlled television set is shown in the next figure.

![Block diagram of an IIC controlled television set](image-url)

Figure 2. Block diagram of an IIC controlled television set.
The IIC bus is a multi-master bus, which means that more than one device capable of controlling the bus can be connected to it (e.g. several microcomputers can be connected to the IIC bus simultaneously). The possibility of more than one microcomputer being connected to the bus means that more than one master could try to initiate a data transfer at the same time. To avoid the chaos that might be ensue from such event, an arbitration procedure has been developed. This procedure relies on the wired-AND connection of all devices to the IIC bus. The IIC protocol also supplies a wait-state mechanism, which is required when a slave has to perform another function (e.g. service an internal interrupt). The number of devices that can be connected to the bus is solely dependent on the limiting bus capacitance of 400pF. The maximum data rate on the IIC bus is 100 kbit/s. Every data transfer is done byte-wise. The number of bytes that can be transmitted per transfer is unrestricted. Each byte has to be followed by an acknowledge bit, sent by the receiver.

The IIC components can be controlled by putting in a master externally. Using this mechanism an IIC based set can be adjusted by an automated system.

II.4 PAPS

PAPS stands for Picture Acquisition and Processing System. It is a modular solution to picture processing problems. Eventually it has been designed to solve visual inspection problems that arose from the need for a high acceptable quality level of components. PAPS has a versatility of applications. A special range of hardware processors has been designed, which causes processing speeds that are unmatched by software processors. But if time is not the most important factor, then PAPS can be programmed to work in software as well.

The basic configuration of the PAPS is shown in the following picture.

---

**Figure 3. Basic PAPS configuration.**

---
As can be seen from the block diagram, PAPS has two bus systems, a picture bus (PI-bus) and a control bus (General Purpose Bus Simplified). These buses provide the data and control signal paths, upon which PAPS is based. The GPBS bus is the same as used in the MOBUS concept. For exchange of data there are three different data channels on the picture bus. The PI-bus also contains several picture related control lines (synchronization, picture enable, line enable, lines to indicate a window).

The PAPS system basically contains three functions: picture data production and monitoring, hardware processing and data transfer, and software processing.

To produce picture data, all PAPS systems have a picture sensor and a picture acquisition unit (PAU). The picture acquisition unit consists of a standard part and a bolt on part for the particular sensor. PAPS assumes the illumination problem has been resolved. For TV cameras an optional extra, a shading compensation unit, can be fitted. This unit ensures that all discrepancies in picture brightness are eliminated. The following sensors, with their sub-unit that fits onto the standard main part of the PAU, are available: a laser scanner, a solid state linear array, a TV camera and a solid state image sensor.

The PAU produces a standardised set of picture control signals and picture data, which it puts onto one of the channels of the picture bus. The PAU is also connected to the GPBS bus for the transport of function control signals to and from the computer. A monitor can be used to display either the real time picture or a processed picture.

Usually, because of high speeds required, hardware processing is used, in conjunction with a computer for post processing and control functions. The hardware processors do the bulk of the work, passing a reduced amount of data to the computer. A variety of hardware processors is available for quick, real time operation. All hardware processors can be regarded as optional. Examples of some processors are:

- Erosion and Dilatation Unit (EDU). This unit is used to make displayed images smaller or larger by eroding or dilating edges. An application of this unit is the removal of all images below a certain size, which has the effect of removing noise from the picture.
- Image Frame Store (IFS). The IFS is used to store one or two digital black and white pictures. It also can be used to permit logical operations between two digital pictures.
- Window Look-up Table (WLT). The WLT is used to check each point in the picture in turn. It is programmable, such that various operations can be carried out on the picture points as required. An application is contour determination.
- Real Time Recognizer (RTR). This can be used to recognise patterns and to detect where and how the objects are placed. The main applications of the RTR are real time contour detection, robotics and contactless measurements.
- Minimum Distance Operator (MDO). The MDO detects distances in pixels shorter than a given value in a black and white picture. Main application is inspection of printed circuit boards and thick film substrates.
After picture data has been processed or, when no hardware processing is required, straight from the PAU the results go to the PTB which realises the transfer of data to the computer. The PTB has a buffer store of 4k words of 16 bits. Data storage and access can be done in several ways (varying order and size). Data flow between the PTB and the computer is possible in only one direction, from PTB to computer.

The software processing is done by the computer, which is also initializing and controlling the PAPS via the GPBS bus. It also receives data from the PTB via the GPBS bus. The application program on the computer will determine what is done with this information and what actions are to be taken. This application program will be working inside the AMTS layer.
The automated geometry adjustment system has to fulfill several requirements. These requirements can be split up into environmental requirements, functionality requirements and accuracy requirements. In this chapter all requirements are formulated.

III.1 Environmental requirements

With 'environmental requirements' all restrictions on the AGAS are meant, that are related to the environment the AGAS is applied in. The main requirements consider the physical environment the AGAS is implemented in, means to realise the AGAS (regarding standardization on equipment), and the maximum costs allowed to build it. These requirements partly have been composed in consultation with people dealing with production; the means to be used have been prescribed by superiors inside the CFT.

The AGAS will be integrated in a flow process. A cycle in this process lasts approximately 30 seconds, from which 5 seconds are used for transport and mechanical handling, so 25 seconds will be available for performing the adjustments. The AGAS will be positioned adjacent to a transport line carrying the devices to be adjusted. System components have to be installed as compactly as possible. Any part of the AGAS has to be positioned within 1.5 m of the transport line.

The production batch size will be at least one hundred. This requires full flexibility of the AGAS. It has to adapt automatically when a new batch arrives on the line. The diameter of the screen of a device on the line can vary between 17 inch and 33 inch. The devices will be positioned in a way, that the vertical axis of the screen (y axis, the x-y plane assumed to be the plane parallel to the transport line and the screen, and perpendicular to the factory floor) will always have the same x position.

The AGAS has to be able to perform all adjustments without being dependent on any other system in the factory. The software has to be based on the software environment called AMTS. Concerning hardware, it has to match the MOBUS concept. It's already possible to use MOBUS equipment in combination with AMTS. When necessary, software has to be written or adapted to suit AMTS for equipment which has not been implemented, yet.

Last but not least, the ultimate capital costs required to build a AGAS mustn't exceed Hfl 50,000.--.

III.2 Functionality requirements

A few functionality requirements prescribe aspects of the functional outline of the AGAS. One of those requirements is that the system will be self starting.

It has to be usable for DUTs of all colour encoding standards (PAL, SECAM and NTSC). Both a composite video signal (CVBS) and an HF signal modulated by the video signal have to be available. The stimulus for adjusting the DUTs is restricted to drive IIC controlled devices. This is chosen because these devices are produced on large scale nowadays; the production line for these types will be considered for automation first. Another reason for
restricting ourselves to IIC is the simplification of the total problem obtained this way.

III.3 Accuracy requirements

Some requirements are formulated concerning the accuracy the AGAS has to have. These can refer to the adjustments made, but can also be related to the accuracy in measuring. A relationship between those two will be discussed.

An investigation to find requirements prescribed by perceptive aspects didn't lead to anything. Obviously, no research has been done on perception of geometry by human beings. Requirements have been formulated in consultation with people dealing with product management and derived from existing adjustment prescriptions.

First the demands for accuracy of the adjustments are discussed. The geometry of a picture can be described by the use of ten parameters. These are parameters that have to be adjusted in an IIC controlled device. However, they can be used to describe the geometry features of any picture, so when other devices are to be adjusted the requirements still will be usable. The parameters meant are:

1) Horizontal shift (horizontal picture position)
2) Vertical shift (vertical picture position)
3) Vertical linearity (equal height of upper and lower half of the picture)
4) Vertical S-correction (equal line distance in top and bottom of picture compared to the central part)
5) Horizontal amplitude (picture width)
6) Vertical amplitude (picture height)
7) EW-parabola (straightness of vertical lines at the left and the right side of the picture)
8) EW-corners (straightness of vertical lines in the corners of the picture)
9) EHT compensation (to stabilize picture size at different brightnesses)
10) Trapezium (equal width of upper and lower half of the picture)

Because some parameters refer to the same part of the picture, they are taken together in the formulation of the constraints. First of all the horizontal shift has to be adjusted in such a way that the horizontal position of the vertical axis of the picture is less than 0.8% of the screen width (0.8%SW) away from the horizontal position of the physical centre of the screen. The vertical position of the horizontal axis of the picture has to be positioned less than 1.5% of the screen height (1.5%SH) away from the vertical position of the physical centre of the screen. The vertical linearity and the vertical S-correction have to be adjusted to achieve the situation that the distance between any set of, theoretically equidistant, horizontal lines doesn't vary more than 2% along the total height of the picture.

EW-parabola, EW-corners and trapezium adjustments have to accomplish a variation in picture width along the total screen height which is less than 0.8%SW.

The picture width has to be adjusted to 48.5µs ± 0.8%SW. The picture height has to be 540 video lines ± 1.5%SH. Line number
167 is the line in the vertical centre of the screen. The EHT (Extra High Tension) compensation has to make sure the change in picture height will be less than 1.5%SH and the change in picture width will be less than 0.8%SW, when the brightness is changed from its minimum to its maximum value.

Now all requirements have been stated for the adjustments, something can be said about the accuracy requirements for the measuring equipment. Obviously the smallest acceptable margins for the adjustments are 0.8%SW and 1.5%SH. The measuring equipment is assumed to work more accurately; this is essential to perform the adjustments. When several measuring results are compared, the errors are added. The absolute maximum number of points to be compared in my situation will be four (when comparing two distances on the screen). For this reason the accuracy requirements for the measuring equipment are set five times as severe as those for the adjustments: horizontally 0.16%SW and vertically 0.3%SH. This leaves margins for the adjustments of at least 0.16%SW and 0.3%SH.
IV CONFIGURATION OF THE AUTOMATED GEOMETRY ADJUSTMENT SYSTEM

In this chapter the total AGAS is configured. Criteria for the choices made are price, performance and ease of implementation. The AGAS will be split up into sub systems. Alternatives are considered for each of these sub systems. Finally a definite configuration is composed.

IV.1 Subdivision of the total AGAS into sub systems

The AGAS can be subdivided into sub systems, based on functionalities required in the AGAS. By doing this, the big problem of developing the total AGAS is split up into several smaller problems concerning the design of the sub systems. Major functionalities are separated, leading to sub systems with reduced functionality requirements.

The automated geometry adjustment system is divided into six sub systems. Each sub system represents a basic task in the total AGAS. These sub systems are:

- Optical sensor. The image appearing on the screen of the DUT has to be inspected by some sort of optical sensor.

- Optical sensor positioning equipment. A wide variety in OUTs can be supplied to the AGAS. In order to obtain sufficient accuracy in measurements, it may be necessary to change the position of the optical sensor.

- Image processing equipment. Information of an image projected on the optical sensor is acquired by this equipment and converted to a format that is acceptable for the computer. A data reduction may take place to reduce the amount of information that has to be transported to the computer.

- IIC stimulus equipment. The initial AGAS will be designed for t.v. sets which are controlled via the IIC bus internally. All data transfer concerning adjustment of geometry parameters will be controlled and done by the IIC stimulus equipment.

- Test pattern generator. In order to get relevant data from the screen of the DUT efficiently, a test pattern generator is used to generate the right patterns at the right moment.

- Computer. The computer executes the overall adjustment program. It controls all other sub systems by sending the appropriate control signals. It also sends data to or receives data from some of these sub systems, depending on the characteristics and functionalities of each sub system.

The next block diagram illustrates the division of the total AGAS into sub systems.
Figure 4. Block diagram of the total AGAS.

As stated, the computer runs the main adjustment program which is the software part in the AGAS. It may be possible though, that parts of the other sub systems are implemented in software as well. So finally, parts of these sub systems may be integrated in the software running on the computer. Among other things, this possibility will be considered in the next paragraph, discussing the functionality requirements of all sub systems separately.

IV.2 Functionality requirements of the sub systems

For each of the sub systems functionality requirements can be formulated. In this paragraph these requirements are considered; when applicable they are visualised by formulating them in block diagrams. The requirements are used as a measure to see whether alternatives are acceptable for this application.

The optical sensor

Not much can be said about the functional requirements of the optical sensor. Actually, there is only one requirement, saying that the sensor has to be able to convert an amount of photo energy into an electrical quantity which is related to the quantity of collected energy in a known way.

The optical sensor positioning equipment

A block diagram, explaining all fundamental functions of the optical sensor positioning equipment is shown in the next figure.
The optical sensor positioning equipment receives data from the computer which supplies information about the required position of the optical sensor. This data is assumed to have a format like \((x, y, z)\). It depends on the situation, how these 'coordinates' have to be interpreted. For instance, the \(z\) may have a spatial meaning but it may also refer to an amount of 'zooming'. It is also possible that one of the coordinates is constant, which means it doesn't have to be sent by the computer at all; in that case the data will be two dimensional.

The received coordinates have to be converted to some variables \((st_1 \ldots st_n)\) representing the physical stimuli needed to get the optical sensor in the right position. Once these stimuli have been calculated, they have to be realised physically. This will cause the mechanical hardware to take action, thus moving the optical sensor. The calculation of the stimuli can be done either by hardware or by software; the creation of the stimuli will always be a hardware function, and so will the mechanics.

The image processing equipment

The basic functionalities of the image processing equipment are shown in the following figure.

A signal supplied by the optical sensor forms the input signal of the image processing equipment. This signal is converted to a binary code to be able to process it in a (digital) computer. This is accomplished by the sensor signal formatter.

A great part of the data describing the image will be worth nothing to the AGAS, because relevant information will be kept in rather small spaces in the image (test patterns are designed so, to limit the amount of relevant information in advance). The data reduction block has to take care that only relevant data is transferred to the computer. It may also transform the relevant information to a more useful format (e.g. contour detection).
The resulting amount of data is stored in a buffer which simplifies the transfer of the relevant data to the main program; the computer will be rather free in choosing the moment of emptying the buffer. The format of the relevant data is not specified yet; it will be a feature of the image processing hardware that is chosen or designed.

It is preferable to realise all three parts in hardware because of the great number of operations that has to be done in as short time as possible. It is possible, though, to reduce the amount of data in software and/or to store the data in a software buffer (e.g. an array). Every part of the image processing system may be controllable by the computer, depending on its realization.

The IIC stimulus equipment

The IIC stimulus equipment receives data from the computer and an address to which the data has to be sent. It has to take care that the right data reaches the right place, in accordance with the IIC rules.

The following picture depicts all functionality requirements of the IIC stimulus equipment. A line connected to all parts, indicates a external controlling possibility.

![Functionality diagram of the IIC stimulus equipment.](image)

Figure 7. Functionality diagram of the IIC stimulus equipment.

First of all the received data and the address have to be fitted in an IIC-data structure. Then the data structure is available to be sent over the IIC bus. The information cannot always be sent immediately, because the IIC bus is a multi master bus; this implies that the equipment has to follow an IIC protocol, and it sometimes will have to wait before sending. The part which guards the IIC protocol is called the IIC protocol handler. When the moment of sending has arrived, the protocol handler supplies the right code to the receiver/transmitter which puts it onto the IIC bus. All parts may be controlled by the computer.

The test pattern generator

The test pattern generator supplies the right test pattern to the DUT under control of the computer. The functional structure of the test pattern generator is pointed out in the next figure.
The test pattern generator receives a set of parameters which, among other things, implies the selection of the right test pattern. The part which performs the choice and the generation of a programmed pattern is called pattern selector.

In order to get a colour signal, a colour encoder is needed. When a modulated HF signal is required, it can be created by the modulator, using the colour or the B/W signal. The choice of colour encoding system has to be controllable by the computer, because a lot of T.V. sets can work on several colour systems. The choice between colour signal and HF signal don't have to be controllable by software; multiple connections on the DUT carrier will enable to choose depending on characteristics of the DUT. It depends on the realization of the AGAS whether the choice of the input signal for the modulator has to be software controllable or is made once for all.

The computer

The computer is situated in the centre of the AGAS. It governs the overall behaviour of the AGAS by executing the test program. It receives data from the image processing equipment; it sends data to the image processing equipment (control), the camera positioning equipment (data), the IIC stimulus equipment (data + control) and the test pattern generator (control). The formats of the data will depend on the features of the several sub systems. The functionality of the software running on the computer has already been described in chapter II when AMTS was introduced. The hardware requirements are dependent on the connections and signals required by the other sub systems. It is certain however, that the computer has to have an interface to the GPBS.

IV.3 Available alternatives for each sub system

In this paragraph for each of the sub systems some alternatives are evaluated, which appeared to be acceptable for this application. Results have to lead to a selection of equipment to be used in the AGAS. Concerning the optical sensor, a choice is made immediately because this is important for determining whether other equipment is usable or not; another sensor would lead to other image processing equipment.
IV.3.1 Optical sensors

Three categories of optical sensors can be distinguished, classified according the number of dimensions in space the sensor supplies information about. These categories are:

1. "Spot sensors": 0-dimensional
2. "Line sensors": 1-dimensional
3. "Plane sensors": 2-dimensional

The first type of sensors only gives information about one point in space. To use this type of sensor to measure geometrical aspects on a screen, time will have to be related to the observation (remember the image is written on the screen by a scanning beam with a known period!). However this introduces an extra quantity (time) in the calculations which, in fact, is absolutely not relevant to the things we want to measure. Another drawback of the use of such a sensor is the fact that the sensor will have to be mounted relative to the screen very accurately. This is very difficult to realise in a cyclic production process and limits the flexibility of the AGAS. More than one sensor will be necessary, because information is needed about several places on the screen. In summary it can be concluded that this type of sensor can not be useful in this application.

A line sensor is a lot more interesting because it gives spatial information as well. This deletes the need for an accurate mounting of the sensor near the screen. The determination of the place of the screen can be done in software and every other spatial measurement can be related to this memorised information. A disadvantage of the line sensors is their restriction to one spatial dimension. In case of geometry, information about two spatial dimensions is required. To obtain such information several line sensors can be used to create a "more than one"-dimensional sensor. It is preferable, however, to observe a complete plane because the place a test object (the screen or a figure on the screen) is appearing will be rather unpredictable. This would lead to the use of a lot of line sensors and high accuracy requirements for the sensors positioning equipment.

Considering the construction of plane sensors we see that, ignoring demanded accuracy for the moment, the total screen can be observed by one sensor. There won't be one place on the screen which is not seen. This makes the plane sensor very likely to be used.

Within the category of plane sensors two main types can be distinguished: the conventional scanning camera and the relatively new CCD camera. When both alternatives are inspected the CCD camera appears to be preferable. It will be very difficult to use a scanning camera to look at a scanning t.v. set. It is very likely that the scanning spots won't match each other leading to a video signal coming from the camera, which contains the information of a black screen.
The price of a CCD camera is lower than that of a scanning camera because it is widely used nowadays (consumers market included); production of these cameras will become cheaper and cheaper. This is also caused by the growing knowledge and experience concerning the production of large scale chips. Furthermore the CCD chip has excellent geometrical features due to the way it is produced. A scanning camera has bad geometrical characteristics; they cannot be corrected simply by calibration in software because they vary continuously due to a dependency on temperature and age of the camera. An outstanding feature of a CCD sensor is its high tolerance to overexposure, which is very important in this case; scanning cameras tend to get burnt in after a while, when looking at the same picture all the time. A disadvantage of the CCD camera is its discreteness in space. This introduces an error, but that may be compensated or decreased by other means.

Philips is producing a CCD camera typed 56470. It contains a CCD chip type NXA1011/01, which has 604(H) x 588(V) pixels. This resolution is the highest available at this moment. The price of the camera is approximately Hfl 1500.--. If the errors caused by the use of a CCD camera will not exceed the acceptable limits, this type will be used in this application. Optics have not been considered yet. This is done later.

IV.3.2 Image processing equipment

A two-dimensional CCD camera has already been chosen to be used as the optical sensor. This camera delivers an analog CVBS signal. For such a camera numerous video acquisition and processing systems are available. Only a few are considered, though, because a lot of systems have too low a performance and others resemble the systems considered.

Three image processing systems are discussed. The first one is an add-on set for an IBM-PC/AT containing a DT2851 (frame grabber) and a DT2858 (frame processor) supplied by Data Translation Ltd. The grabbing is done with a resolution of 512*576 for the European video system. The processing is not done in real time (i.e. the data is buffered and processed on a slower speed than it is produced); frame averaging takes 0.5s and a 3*3 convolution (e.g. for edge detection) takes 2.0s. The set costs approximately Hfl 25000.-- including all software.

Another image processing system from Data Translation Ltd consists of a DT1451 (frame grabber) and a DT1458 (frame processor). This is a system similar to that mentioned above, except that it is a little faster (averaging takes 0.3s, 3*3 convolution 1.4s), and it fits in and is controlled by VME. The price of this system also is approximately Hfl. 25000.-- including all software.

Last but not least, there is the PAPS (Picture Acquisition and Processing System), supplied by the Centre For manufacturing Technology (CFT) of Philips Eindhoven. PAPS has a resolution of 780*576 pixels for the European t.v. system and it processes every picture in real time (pixel processing is done at the same speed as pixel production, no buffer required). It is a modular system in which one can choose the right card to perform the right function. For the actual application a VAU (Video Acquisition Unit, this is the CCD camera-version of the PAU), a PTB (Picture Transfer Buffer) and a RTR (Real Time Recognizer) will
be sufficient. The PAPS is controlled via the GPBS, which is also used by MOBUS equipment. The PAPS cards mentioned, will cost about Hfl 14000.--. Software will be supplied fairly cheap or will be self-written (not very difficult because of a known specification and specialization inside the CFT).

IV.3.3 IIC stimulus equipment

For the IIC stimulus equipment two alternatives are available. The first one is a card called IIC-20, designed to fit in the MOBUS concept. It is controlled by the GPBS bus. Its operation is simple; all required functions are accomplished in hardware. The card itself takes care of the IIC protocol. Its price is Hfl 2400.--. Again, software will be supplied fairly cheap or will be self-written.

Another alternative is an add-on for the IBM-PC which has been designed by ELCOMA. It is not a product in accordance with the requirements and quality standards as applied for MOBUS cards, but it is reliable and functions very well. The price of the card is less than Hfl 1000.--. The add-on itself takes care of the IIC protocol. Software for controlling it is partly available.

IV.3.4 Test pattern generator

The generation of a test pattern can be done with the help of two acceptable alternatives. One is a card called VTG-20, fitting in the MOBUS concept. It is a programmable Video Timing Generator; with the help of a software tool called VTG-SOFT the user can define his own test patterns. These patterns are stored in a battery backed-up RAM. The card can create pictures for all kinds of t.v. systems. To obtain a colour picture according to a particular system, an additional card named VxE-20 is needed (Video Encoder, x=P for PAL, x=N for NTSC, x=S for SECAM). These additional cards are connected to the VTG-20 by a separate flat cable. Control of the VTG-20 takes place via the GPBS; the encoders are controlled via the VTG-20. The price of a VTG-20 is Hfl 5000.--, a VxE-20 costs approximately Hfl 3000.--.

When a modulated HF signal is required, a modulator has to be used. This modulator can be realised using components from the Electronic Distribution System supplied by Philips. Using these, a modulator can be composed according to any preferred colour encoding system, with the test pattern put on any channel within the VHF or UHF band. Similar systems are already used in all kinds of applications. People who have experience in using these modules are very pleased about the performance and the reliability of the EDS system. The price of such a modulator will be Hfl 1350.--.

Another programmable test pattern generator is the cheapest one from the PM563x series, PM5631, supplied by Philips. Test patterns can be programmed in ROM. The generator delivers a PAL signal and contains an HF modulator. It is controlled via the IEEE bus. The price of such a piece of equipment is Hfl 9000.--.
IV.3.5 Computer

The alternatives for the computer were already known as the 68000 VME system on one hand and the IBM-PC on the other hand. The 68000 VME system is supplied by Philips I&E. It costs approximately Hfl 25000.--. This includes a processor board, a 20 Mb Winchester, a floppy disk drive, a disk controller card, 1 Mb DRAM, a VT220 terminal, mounting tools, and a combination of a translator/bus controller (PG2910) and an EXPA, which is a card that translates the VME bus to the GPBS. The EXPA is included because the GPBS system is the only viable instrumentation system to be connected to the VME bus in this situation. An IBM-PC is supplied by a lot of companies, including Philips. The price of an AT-compatible computer from Philips (PM3202) is Hfl 6000.--. This computer contains a 20 Mb Winchester, a 1.2 Mb floppy disk drive, 640 kb RAM and an enhanced graphics colour monitor. To control the GPBS a bus translator set is required which has been developed at the CFT. This bus translator set is called IGC-00. It consists of two cards: an add-in card for the computer (IGC-00/S) and a bus driver card (IGS-20); its price is Hfl 1000.--. To control an IEEE instrument another bus translator is needed, which will probably cost about Hfl 2000.--.

IV.4 Composition of the AGAS

Knowing the available alternatives, several systems can be composed using varying ingredients; this is done eight times. From these eight configurations one is chosen to be used. The camera and its optics are not accounted for in all options, because this is the same for every configuration; neither is the positioning equipment. Prices do include all required mounting facilities.

**Configuration 1**

<table>
<thead>
<tr>
<th>Computer</th>
<th>Vision system</th>
<th>Pattern generator</th>
<th>Stimulus equipment</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>68000 VME</td>
<td>VAU + PTB + RTR</td>
<td>VTG-20 + VxE-20 + EDS</td>
<td>IIC-20</td>
<td>Hfl 53750.--</td>
</tr>
</tbody>
</table>

**Configuration 2**

<table>
<thead>
<tr>
<th>Computer</th>
<th>Vision system</th>
<th>Pattern generator</th>
<th>Stimulus equipment</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>68000 VME</td>
<td>DT1451 + DT1458</td>
<td>VTG-20 + VxE-20 + EDS</td>
<td>IIC-20</td>
<td>Hfl 64750.--</td>
</tr>
</tbody>
</table>

**Configuration 3**

<table>
<thead>
<tr>
<th>Computer</th>
<th>Vision system</th>
<th>Pattern generator</th>
<th>Stimulus equipment</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM-PC/AT</td>
<td>VAU + PTB + RTR</td>
<td>PM5631</td>
<td>ELCOMA add-on</td>
<td>Hfl 36000.--</td>
</tr>
<tr>
<td>Configuration</td>
<td>Computer</td>
<td>Vision system</td>
<td>Pattern generator</td>
<td>Stimulus equipment</td>
</tr>
<tr>
<td>--------------</td>
<td>---------</td>
<td>---------------</td>
<td>-------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td><strong>4</strong></td>
<td>IBM-PC/AT</td>
<td>DT2851 + DT2858</td>
<td>VTG-20 + VxE-20 + EDS</td>
<td>IIC-20</td>
</tr>
<tr>
<td><strong>5</strong></td>
<td>IBM-PC/AT</td>
<td>DT2851 + DT2858</td>
<td>PM5631</td>
<td>ELCOMA add-on</td>
</tr>
<tr>
<td><strong>6</strong></td>
<td>IBM-PC/AT</td>
<td>VAU + PTB + RTR</td>
<td>VTG-20 + VxE-20 + EDS</td>
<td>IIC-20</td>
</tr>
<tr>
<td><strong>7</strong></td>
<td>IBM-PC/AT</td>
<td>VAU + PTB + RTR</td>
<td>VTG-20 + VxE-20 + EDS</td>
<td>ELCOMA add-on</td>
</tr>
</tbody>
</table>

**Configuration proposal**

Looking at these configurations the following proposal was made for the configuration to be realised:

<table>
<thead>
<tr>
<th>Computer</th>
<th>Vision system</th>
<th>Pattern generator</th>
<th>Stimulus equipment</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM-PC/AT</td>
<td>VAU + PTB + RTR</td>
<td>VTG-20 + VxE-20 + EDS</td>
<td>ELCOMA add-on</td>
<td>Hfl 45350.--</td>
</tr>
</tbody>
</table>

This configuration has several advantages. It is relatively cheap, thus being economical attractive. This is mainly caused by the cheap computer used. The use of an IBM-PC is admitted because
a huge data reduction can take place on the PAPS, which causes a short transfer time for the reduced amount of data from PAPS to the IBM-PC. Normally it would cost a lot of time (several seconds) to transfer data of a total picture to the PC, and then we are not even talking about processing time required. All picture processing is done real-time and accurately. Knowledge about all used instruments is present inside Philips. An integration of two instrumentation standards produced by the CFT, PAPS and MOBUS which were always used separately until now, is realised; the software of these standardized systems will be integrated as well. Furthermore this configuration fulfills the customer's requirement of low-cost computer technology to be applied in a structured way.

IV.5 The definite AGAS: a detailed description

In this paragraph a detailed description is given of the instruments used to realize the sub systems. Only the IBM-PC/AT is not described because this is a commonly known instrument. Describing it would take too much effort as well, while lots of documentation are available already.

IV.5.1 Philips CCD camera type 56470

The Philips 56470 camera is a black and white CCD camera containing the NXA1011/01 image sensor. This sensor yields a resolution of 604(H) x 588(V) pixels. It is a Frame Transfer Sensor. According to the frame transfer principle, each field (pixel) of the complete picture frame is separately integrated within a photosensitive imaging region, transferred by CCD shift registers into a storage region during vertical blanking, and then clocked out serially to form the video signal during the subsequent field integration period. The chip has been split up into periodic groups of 3 columns; this is done to be able to use the production process of the B/W chips as a part of the process to produce the (almost identical) colour chips. The chip has three outputs representing column number (3*k+1), (3*k+2) and (3*k+3), 0 < k < 200. The clocking out is done by three 3.85MHz shift clocks with a phase difference of 120 degrees, resulting in a readout of 11.55 million pixels per second. The camera delivers an analogue video signal (CVBS) according to the CCIR specifications. To convert the three pixel streams into one analogue video signal a multiplexer and a sample and hold circuit are used.

An external sync lock facility has been implemented in the camera. On this input active low sync pulses conform the CCIR specification can be supplied for controlling the synchronization of the camera.

A video amplifier in the camera has a controllable gain, which can be adjusted externally by applying a dc voltage between 0 and 11V. When no voltage is applied the camera goes in the automatic gain control mode. The camera also contains an iris control driver to drive electronics in a lens which adjust the iris in such a way that the light level on the sensor is always optimal. Display equipment, like t.v. sets, sometimes have a signal distortion called gamma distortion. The camera contains a gamma
correction circuit which gives a predefined signal distortion inverse to the gamma distortion so the resulting picture should be equal to the original one. The gamma correction is made switchable. Via a scan mode input one can choose whether the output signal will be interlaced or non-interlaced.

IV.5.2 Picture Acquisition and Processing System

Because of the use of a CCD camera, the functional layout of the image processing equipment that has been designed as shown in IV.2 is changed to the one depicted in the following picture. The optical sensor signal is an analogue CVBS signal which is converted to digital data by an A/D converter.

![Functional block diagram of image processing equipment](image)

Figure 9. Functional block diagram of image processing equipment.

The way the functional requirements, shown in the block diagram of the image processing equipment, can be met is explained in the following sections.

A/D Conversion

The A/D conversion is done by the Video Adaptor, which is a part of the VAU (Video Acquisition Unit). The type of this VA depends on the type of optical sensor that is used, in this case a two-dimensional CCD camera.

The VA has several additional features. A block diagram explaining the VA is shown in the next figure.

![Block diagram of a PAPS Video Adaptor](image)

Figure 10. Block diagram of a PAPS Video Adaptor.

The VA contains a switch to select the input signal from four camera inputs. The selected camera signal is connected to a clamp circuit (which acts as a buffer and offset corrector for the ADC)
and to a sync separator. The ADC (Analog to Digital Converter) provides the digital data for the Acquisition Unit. The sync separator produces the CS (Composite Synchronization) signal for the Acquisition Unit.

The LS (Line Synchronization) signal synchronises the 15MHz oscillator and the monitor display. The 15MHz oscillator produces the GCK (General Clock) pulses for the PAPS.

The Digital Video data (DA1-DA8) from the acquisition unit is converted to analogue and mixed with the DSPL (display line) signal. This combined video is mixed with the LS signal and connected to the monitor.

The VA cannot act by its own; it has to be used in combination with the Acquisition Unit. This AU is required to interface the VA to the rest of PAPS. Besides it has some additional features that can be very useful. A block diagram of the AU is shown below.

![Block diagram of the Acquisition Unit](image)

Figure 11. Block diagram of the Acquisition Unit.

The digital picture data (AD1-AD8) from the VA is connected to a video threshold circuit and the PI-bus output selection circuit. The video threshold circuit uses data from the control registers to determine the threshold level. This level is used as the borderline in making binary pictures. The digitized data is leaving the VAU via the PI-bus output selection circuit. Here a choice is made between the DIG VID input (a binary picture, black and white) and a grey level picture. Then the selected picture is put onto one of the three data channels of the picture bus.

The data for the sub-sampling circuit (which controls on what part of the 15 MHz frequency the picture data is sampled) and for the window and cursor generators comes from the control registers, which are filled by the controlling computer. The generation of a window can be used to restrict operations to a certain part of interest on the image. The cursor can be used to point at things; it will be visible on the monitor.
The input control data circuit is used to switch various programmable data control signals to the window and cursor generators. The output control data circuit is used to switch various data control signals to the picture bus.

The PI-bus input selection circuit selects data from one of the three picture bus data channels and connects it to the VA for the monitor display. The sync separator accepts the CS (composite synchronization) signal, which it separates into various signals required by the V AU. The CS signal is also buffered and set as inverted COMPSYNC onto the picture bus.

Data Reduction

Data reduction is a preferable feature of the image processing equipment because of the huge amount of picture data that has to be processed. To give an idea of the amount: when the video signal is sampled on 15MHz and one frame contains 270 tv-lines then one frame will produce the information of approximately 200,000 pixels. When each pixel takes $2.5\mu\text{sec}$, the processing of one frame already takes 0.5 second! Several alternatives are available for data reduction. All alternatives pick up the digital image information that is put on the PI-bus by the V AU.

A possibility to reduce the amount of data is the use of a Real Time Recognizer (RTR). This is a hardware processor card which performs real time recognition on black and white video pictures, by means of one or two reference pictures of 21 columns * 12 rows or one reference picture of 21 columns * 24 rows. The restriction to a black and white picture is made to increase speed; now the detection of pictures can be done by just looking at the most significant bit of the grey value byte. The RTR performs the recognition in real time, as indicated by its name.

The RTR for instance can be used for contour determination. To detect all contours in an image, two reference pictures have to be created; the first contains a black to white transition, the second a white to black transition.

Generally, a threshold can be set for the minimum number of matching pixels to be acknowledged as a recognition. It is also possible to add or subtract pixels onto the contour of objects of the incoming image. The incoming information can be subsampled by the RTR with an adjustable factor between $1/127$ and $127/127$.

The result of the real time recognition will be a binary (B/W) picture.

Another possibility to reduce data is the use of the Picture Transfer Buffer. This card always has to be used to interface the PAPS to the computer. It has several features to reduce the amount of data; in a certain mode the card only registers the coordinates and/or the grey value of the pixels which have a most significant bit equal to one. In the case, where the incoming image contains very few pixels with a grey value which is higher than half the maximum grey value, the PTB can be used to select only the information of these particular points to be transferred to the computer. Another possibility is to store only the MSB of each grey value byte.

The preferable means for data reduction depends on the test pattern(s) to be used. For this reason, no choice can be made right now.
Relevant Data Buffer

As said, to transfer data from the PAPS to the computer, the Picture Transfer Buffer card is required. The data can be produced by the VAU, but it can also be a result of one of the hardware processors. The PTB card contains an internal FIFO (First In First Out) buffer store that holds data waiting to be transferred into the computer.

A block diagram of the PTB is shown in the next figure.

Figure 12. Block diagram of the Picture Transfer Buffer.

The input data selector selects one of the three data channels on the PI-bus. The data valid logic logically combines the control signals from the PI-bus with the MSB from the selected data channel to produce a data valid signal which is fed to the packer circuit. The packer circuit packs incoming data into 16-bit words conform the operation mode of the PTB. The input channel width can be selected to be 1, 8 or 16 bits. Once the data has been packed, a Data Ready (DR) signal is sent to the RAM control circuit. This circuit is connected to a 4k*16 RAM buffer store; it generates all required signals for the RAM.

Counters for x and y positions have been implemented, such that points of interest, which may be detected by one of the processors, can be stored, together with their x and y coordinates, depending on the operation mode of the PTB. The y coordinate is always stored once per line, during the line synchronization pulse. Pointers for reading from and writing to the memory select the right addresses; these pointers are auto-incrementing.

The unpacking circuit governs a byte-by-byte or word-by-word transfer of data to the computer. Transfer of data can be either sequentially by means of data blocks or randomly. In the first case, the transfer is controlled by the I/O processor; in the second case by the computer program.

IV.5.3 Philips M0dular BUilding System

Next figure once again depicts the functional requirements of the test pattern generator.
The pattern selector has been implemented in a MOBUS card called VTG-20. It is a RAM based video pattern generator used in test and measurement equipment for video applications. In addition to the pattern selection mechanism it contains several possibilities to adapt the (originally) programmed video signal. The VTG-20 is controlled via the GPBS; by sending data to the module via this bus different patterns and functionalities can be selected. Several video patterns are stored in the RAM library; pattern flexibility is easily accomplished by having computer control over the contents of all RAM's. The VTG-20 can be loaded with pictures with the help of an available software package called VTG-soft. With this package it is also possible to design video lines and pictures; these pictures and lines can be either standard (according to a certain video standard) or non-standard. As well as providing video information the VTG-20 also generates all control, synchronization and blanking signals required by any external colour encoder module. Output signals on the VTG-20 are:

- R, G, B and composite video (B/W)
- composite blanking
- composite synchronization
- horizontal drive
- vertical drive

The parameters for the VTG-20 to adapt the selected pattern signal partially, are:

- Select external Y input (determines whether an external Y signal is added to the composite video signal or is switched off)
- Y filter (Y filters on/off on encoders)
- Amplitude burst (percentage)
- Amplitude chroma (percentage)
- Video polarity
- Composite synchronization polarity
- Vertical drive polarity
- Horizontal drive polarity
- Clock source (internal/external)

For creating a colour signal a separate colour encoder is required. The VTG-20 can control one encoder at a time. It has a mechanism to select one encoder out of four. Every encoder has two composite video outputs and an external Y signal input. The two video outputs are directly connected, and switched to the
encoder circuit when that encoder is selected. This is used to connect several encoders to one line.
In the MOBUS range several colour encoders are available: VPE-20, VNE-20 and VSE-20. The VPE-20 delivers PAL signals, the VNE-20 produces NTSC signals and the VSE-20 creates SECAM signals. All cards are passive MOBUS cards, which means they have no MOBUS interface apart from power supplies. The encoder control signals are derived from the VTG-20 via a flat cable connection on the front of the cards. Each encoder provides all video information according the encoding system specified.
An HF modulator will be built using components of the Electronic Distribution System, supplied by Philips. The EDS contains modules from which a modulator can be composed according to any preferred colour encoding system, with the test pattern put on any channel within the VHF or UHF band. The way a modulator can be configured is shown in the picture below.

![Figure 14. Configuration of a video modulator using EDS.](image)

The incoming CVBS signal modulates an IF carrier in the IF modulator. The frequency of this carrier depends on the colour encoding system required. The type number of the available IF modulator is LHC8010/xx, with xx equal to 45 for NTSC, 55 for PAL B/G and 65 for SECAM signals. The resulting IF signal then goes into a converter from the LHB6090/xx series, which converts it to any VHF or UHF channel.
The nominal output level of the modulator is 85 dBpV; this will be sufficient to feed a television set. The bandwidth of a modulator as shown is 5 MHz for PAL, 6 MHz for SECAM and 4.2 MHz for NTSC.

### IV.5.4 Elcoma add-on for driving the IIC bus

The add-on IIC interface from ELCOMA has been designed to control an IIC bus using an IBM personal computer (or compatible). The interface is connected to the standard Centronics parallel printer interface of the personal computer.
To recall the functionality requirements, the block diagram of the IIC stimulus equipment is shown again in the next figure.

![Figure 15. Functional block diagram of IIC stimulus equipment.](image)
The IIC data structure formatting and all control is done in software. Best is to realise this in an instrument driver. A routine for basic I/O has been written already. This routine doesn't take care for the data structure formatting. Both the IIC protocol handling and the receiving and transmitting are accomplished by the interface hardware.
Several errors will be introduced in the AGAS, due to various causes. It is preferable to know the magnitude of these errors to be able to judge whether the AGAS will function within a sufficient range of accuracy.

In this report the total error of the AGAS will be split up into the errors made in horizontal direction and those made in vertical direction. These errors will be subdivided into separate errors, each originating from a different source. The magnitude of the errors will be estimated or calculated where possible.

V.1 Error causing parts in the AGAS

Errors can be caused by the following parts of the automated geometry adjustment system:

- Test pattern generator (VTG-20)
- Colour Encoder (VSE-20, VNE-20, VPE-20)
- Modulator (Electronic Distribution System)
- Television set
- CCD camera
- Picture Acquisition and Processing System (PAPS)
- Computer

Each part will be discussed in relation to the errors it is introducing.

V.2 Errors made in horizontal direction

The errors made in horizontal direction are significantly different from those made vertically. This is caused by the way both a video camera and a DUT are scanning the picture area. This paragraph deals with all errors made horizontally.

V.2.1 Test pattern generator

The VTG-20 has a finite resolution in both time and amplitude (minimum step in timing is 100ns, minimum step in amplitude is 4mV). This fact, however, will cause no errors. It only causes the programmed test pattern to consist of known steps in magnitude.

The video signal transition time of the VTG-20 is approximately 100ns. This means that each step in the programmed test pattern will result in a transition between two levels in the video signal which lasts approximately 100ns. This transition time can lead to an error in the calculated points of time (a delay). In practice, the transition appears to be approximately linear. When programmed level changes occur every 100ns, the resulting video signal will be a smooth waveform. This fact can be taken into account while designing the test pattern(s) and calculating the points of interest. The test patterns have to be designed causing a predictable behaviour of the video signal. This can be done by programming steps, having all the same size, every 100ns. The result will be a (nearly) linear waveform having a delay of
100ns, compared to the programmed points in voltage-time plane. This delay can be corrected by shifting the programmed test pattern 100ns to the left. The programmed pattern and the delayed output signal are depicted in the following figure.

![Diagram showing theoretical and practical test pattern wave form.](image)

**Figure 16. Theoretical and practical test pattern wave form.**

When a correction is implemented as proposed the error caused by the transition time will be negligible compared to other errors. It also deletes the time discrete character of the test patterns. The inaccuracy in amplitude is less than 2% for the Y signal and less than 5% for the R, G and B signals. The error in the Y signal is caused by the inaccuracy in amplification of an amplifier stage on the VTG. This amplifier is very linear, though. From this point of view it is best to base all measurements on relative behaviour in the test pattern (i.e. the result of the measurements must not be dependent of absolute grey values or intensities). Another advantage of doing so is that the measurements are not influenced by t.v. adjustments concerning luminance (which influence the absolute behaviour) anymore. For instance, a symmetrical figure in the test pattern, with a grey value pattern varying linearly in space (on the screen) or time (in the video signal), can be used to improve the determination of a certain point of interest. The point of interest could be determined by calculating the average value of the coordinates of the points, on which the test figure passes a certain grey level. By averaging, the dependency on absolute behaviour is deleted. To explain this idea, it is represented graphically in the following picture. This figure depicts a one dimensional situation, but the test figure can behave this way in two dimensions as well.
Obviously, the 2% error has no influence on the accuracy of the measurement now. The relative accuracy between several levels in the Y signal is far more important in this case. This depends on the accuracy of the D/A converter on the VTG; its relative accuracy is 0.1% for all possible output levels. The influence of this inaccuracy can be said to be negligible. To omit quantizing errors which may increase the total error considerably, the programmed levels have to match the levels prescribed by the digital representation accurately. In the VTG-20 a unit step in the binary grey value representation results in a 4mV step in the video signal. Therefore the programmed voltages have to be multiples of 4mV.

The bigger number for the inaccuracy in colour signals is caused by the way they are generated on the VTG-20 (matrices). For more accurate colour video signals (R, G and B) an encoder has to be used.

The DC offset voltage in Y signals will be less than 10mV. But when only relative behaviour counts, as in the method proposed earlier, the presence of this offset voltage won't have any influence on the measuring results.

Summarizing, the VTG-20 won't cause any error (out of some negligible ones) when the right test pattern(s) and the right measuring method are applied.

V.2.2 Colour encoder

The bandwidth of the VNE-20 and the VPE-20 is 5MHz (± 0.2dB). This is equal to or even bigger than the theoretical bandwidth of the t.v. set working on the particular system. The VSE-20 has a bandwidth of 6MHz, which equals that of a theoretical SECAM set. So the bandwidth of any of the encoders won't introduce an error. The DC amplitude inaccuracy is less than 1% for Y and 5% for chroma. However, for the same reasons as for the VTG-20, the inaccuracy in Y doesn't lead to any error. Again, the relevant error in Y for us is caused by the relative inaccuracy in output levels of the D/A converter and is negligible.
The 5% inaccuracy in chroma is not important in this case. What really counts for the quality of the picture on the screen is the inaccuracy of the ratio between chroma amplitude and burst amplitude. This inaccuracy is very small (approx. 0.1% according to the designers). Such inaccuracy will only result in a colour which is slightly different. However, the difference will not be noticed by a black and white camera (the use of colours in the test pattern can be preferable for other purposes).

The encoders receive the programmed information from the VTG digitally, so there will be no influence of the VTG bandwidth (or transition time) when a colour encoder is applied. Due to the digital programming of the test patterns, the encoders will have a limited resolution in their Y and Chroma signals, but this does not lead to any error; it is just a limit in possible amplitudes for the test patterns. Again, the levels programmed numerically have to match the levels prescribed by the digital representation accurately, in order not to introduce quantizing errors.

All encoders contain a switchable Y-filter with a -6dB bandwidth equal to the theoretical bandwidth of each t.v. system. In the AGAS these filters haven't got any purpose. But switching on the filters causes an increase in video transition time from less than 30ns to at least 200ns, and a serious timing difference between Y and Chroma, so it is better to have them switched off. In that case the timing error made in the test pattern is 30ns at most.

V.2.3 HF modulator

The only way the modulator can introduce errors is by its limiting bandwidth. But the bandwidth of the EDS modules is sufficient for all video systems. The bandwidth of the LHC8010/xx is 5MHz for PAL, 4.2MHz for NTSC and 6MHz for SECAM. The bandwidth of the LHC6090/xx is a little smaller than theoretically required, approximately 9MHz, positioned symmetrically around the carrier frequency. This isn't a problem though, because the bandwidth of a t.v. set will never exceed 4MHz in practice, as we shall see.

According to people, who used these modules in other test applications where accuracy and stability are of great importance, the performance of these modules is very good and no errors worth mentioning will be introduced by them.

V.2.4 T.v. set

The video signal path in a t.v. or monitor consists of several distinguishable parts that can influence the transformation of the incoming video signal to the picture on the screen. A result of the geometry driving signals, electronics and end stages (including the deflection coils) is the geometric deformation of the picture. However, this error is not accounted for because this is the influence which is to be adjusted by the AGAS!!

Other influences can be caused by noise in the DUT and bandwidth limitation. The influence of noise on the video amplitude will be negligible when multiple points on the grey value pyramid (as
proposed in V.2.1) are used in a position determination. The influence of noise on the timing ('jitter', which is only relevant in horizontal measurements) can be made negligible by measuring a grey value pyramid on several video lines and then averaging the result.

The video bandwidth, however, has to be taken into account seriously. Theoretical bandwidths have been defined for each television system. These are 4.2MHz for NTSC, 5.0MHz for PAL B/G and 6.0MHz for SECAM systems. In practice, the bandwidth of a DUT will be smaller. This is caused by a notch filter (band elimination filter) in the receiver that suppresses the sub-carrier wave(s) implemented for colour purposes. For NTSC the sub-carrier frequency (and thus the centre frequency of the notch filter) is 3.579MHz and for PAL B/G it is 4.43MHz; the sub-carrier frequencies for SECAM are 4.250MHz and 4.406MHz. Because the notch filters are not perfect, the frequency components between the centre frequencies of the notch filters and the theoretical bandwidth of the systems will be attenuated considerably. Practically, this means that the video bandwidth of receivers of any system will be lower than the centre frequency of a notch filter in the receiver. The practical bandwidth of a NTSC set is approximately 3.2MHz; a PAL B/G set and a SECAM set both have a bandwidth of approximately 4.0MHz. These bandwidth limitations all refer to the Y signal and are only relevant to t.v. receivers. Some monitors don't have such a severe bandwidth limitation because they are not driven by HF or CVBS signal but by RGB; for flexibility purposes however, the AGAS has to be designed for the worst case.

Both the theoretical (with an ideal notch filter) and the practical situation are visualised in the following picture for PAL and NTSC. In the SECAM system the two carrier waves lie very close to each other, so the result will be practically the same.

![Theoretical and practical frequency response of a t.v. receiver.](Figure 18)

The chroma signals have a bandwidth which is a lot smaller. For NTSC and PAL B/G the chroma bandwidth is 1.3MHz, for SECAM it is 0.5MHz. This means that the test pattern mustn't contain frequency components beyond 0.5MHz containing colour information, or beyond 3.2MHz containing B/W information. When these constraints are met, the AGAS will be applicable to all video systems and achieve a minimization of errors due to bandwidth limitations.
When these constraints are not met the DUT causes a delay. When the filtering characteristic of the DUT is assumed to be of first order ([1]), the delay caused by this filter can vary between 0 and approximately 100ns, depending on the test figure used and the way the point of interest is calculated. This delay can be calculated rather accurately so it can be corrected very well. This way no errors worth mentioning will be caused by the DUT bandwidth.

Summarizing for the test pattern generator, the colour encoders, the modulator and the DUT, the delay time to correct in software and the error made depend on whether a colour encoder is used or not. When no colour encoder is used, a timing correction is required in the test pattern software; no considerable timing error is made. When a colour encoder is used the delay time is 30ns at most; a correction of 15ns will limit the error made to ± 15ns. In that case no correction can be applied in the test pattern because of the timing resolution of the VTG; it has to be implemented in the test program.

V.2.5 CCD camera

The camera used is a Philips camera, type 56470. It contains a frame transfer CCD with a resolution of 604(H) x 588(V) pixels. The camera can introduce several errors in the AGAS. First of all there is its bandwidth limit which may cause errors. Another error can be caused by the digital character (in time, not in amplitude) of the camera. Geometrical variances on the CCD chip and in the optics can also result in an error. Last, but certainly not least there is the combination of the space discrete character of the CCD chip and the shadow mask in a colour t.v. receiver, causing so called moire patterns.

According to the technical specifications of the camera the -6dB bandwidth is equal to 5.8MHz. The pixel generation rate is 11.55 million pixels/s. Knowing this, Nyquist tells us, that the highest frequency possibly observable will be 5.78MHz. Both the Nyquist and the technical bandwidth stretch far beyond the video bandwidth of the DUT; at 4MHz the attenuation of the signal by the camera will approximate 0dB so the bandwidth restriction of the camera is not causing any error related to the test pattern to be processed. As will be seen later, the bandwidth limitation in combination with the discrete character of the CCD will introduce an error, however.

"The word moire comes from the name of a silk fabric which, when folded, exhibits patterns of light and dark bands."

(A.J. Durelli/ V.J. Parks, Moire analysis of strain).

Moire is caused by two transparent gratings superimposed on each other, where light is sent through. When the incoming light is homogeneous, the resulting beam of light has an intensity varying in space. No extensive explanation of moire will be given here; for more information I'm referring to the vast amount of literature that is available on this subject.

Because the CCD consists of adjacent light sensitive areas (pixels), from which the collected energy (light) is integrated,
the CCD can be seen as a periodic grating. The shadow mask in a
colour television receiver causes columns of one colour (red, green or blue) on the screen of an in-line CRT (which is commonly
used nowadays). This can be seen as another grating, which is
also periodic. Normally the camera will be focussed on the t.v.
screen, so the screen will be projected on the CCD very accurate­
ly. This creates a moire situation; all light coming from behind
the screen (cathode rays) is "filtered by a moire filter". This
causes an amplitude modulation (pixel dependent) of the video
information, which may lead to an error in the AGAS. It is not a
systematic error; the moire pattern is unpredictable. It depends
on the shape of the screen, the form and quality of the shadow
mask, camera positioning, etc.

I have done some tests to gain some insight in the moire effect
in my situation. The moire effect appears to be independent of
the intensity of the incoming light. It is dependent on the
colour of the test pattern. This is caused by the fact that, when
one or two out of the periodical bar of three colour columns on
the screen are relatively dim, there is a stronger amplitude
modulation in the moire pattern compared to the situation when
all three bars are lightened equally. The moire effect appears to
be greatest when the colour of the test pattern is green, red or
blue; in these cases only one colour column is light, the other
two being totally black. When the pattern is white there is
still some moire, due to the fact that the blue column is darker
than the red or green column. From this point of view, the best
colour to choose for the test pattern would be a colour between
white and blue.

It is hard to say what error the moire effect can cause. It may
be possible to delete a major part of this error by considering
grey values instead of just a threshold passing, averaging the
moire influence out of the result. For instance, when a pattern
whose grey value is a pyramidal function of time is used, an ap­
proximating algorithm (e.g. least mean square) can recover the
pyramidal behaviour. In fact this will be a digital filtering,
suppressing moire. The obtained function then can be used to
calculate the point of interest, which is the centre of the
pyramid. Another way to omit moire is not to focus the camera on
the screen. This will cause no spatial periodic image on the CCD
anymore. It will delete moire but may introduce other error(s); this
needs some investigation.

However, I will try to estimate the error made horizontally in
the situation where a position is determined from the places
where a threshold is passed (decision is made by PAPS), and the
camera is focussed on the screen. The figure in the test pattern
that is used to calculate a point of interest is made symmetrical
around that point. The moments of passing a threshold are used to
calculate this point of interest. In order to make the estimation
easier, some things are assumed that simplify the problem.

First of all, I assume a t.v. screen to be completely flat and
positioned parallel to the CCD surface. I also assume that the
screen contains 800x3 columns (which is not very far from reali­
ty), and that the screen is projected on the CCD, filling this
image for 80% horizontally and vertically. This means that the
ratio between the projected t.v. grating and the CCD grating is
1000:600 = 1.67:1. Thanks to the screen being flat, the moire
pattern is a set of parallel vertical lines. A period in this
moire pattern lasts 5 "t.v. columns" or 3 "CCD columns". When a
test pattern segment is symmetric and the centre of it has to be calculated, based on the moments of passing a certain threshold level, the error due to moire in this calculation is always less than one moire period, i.e. 3 pixel periods = 260ns. However, the moire period is very unpredictable. In an experiment it appeared that the modulation varies very smoothly and the modulation index is always less than approximately 0.2 . When a test pattern containing a grey value pyramid is inspected, it seems that, for some reason, there is just a very little influence of moire; I estimate the maximum error introduced to be less than 25ns (0.05% of a line cycle).

It is preferable to make the slope of light intensity in the test figure steep enough to guarantee that the result of the "moire filtered" pattern always has the same derivative as the theoretical signal on the same place on the hill (otherwise the threshold will be passed more times than expected).

The moire effect can not be used in the AGAS as a benefit. Generally, it only gives reliable information about the two gratings producing it. A little information is given about the colour of the incoming light in this case, but that is not of any interest at all. Moire also supplies a measure of the amount of focussing of the camera on the DUT; this may be applicable in a later stage.

Altogether it is preferable to delete the existing moire influences as much as possible. This can be done by defining a figure in the test pattern, for instance a pyramid, with steep enough slopes in intensity. Furthermore, several points on the slope have to be registered, including their grey value. From these points, a best matching straight line can be calculated for both sides of the "grey pyramid". From these lines the original dot can be reconstructed and the centre of it can be determined rather accurate, deleting a great part of the moire influences when enough points on the hill are known.

An advantageous fact of this method is the correction of another feature of the camera, that was already announced earlier. The output signal of the camera theoretically would consist of steps, because of the discrete character of the CCD. In practice, due to bandwidth limitation the transition from one level to the other takes about half a pixel clock cycle (44ns). When the method mentioned above is not applied, the error caused by this fact can be minimized to at most a quarter of the CCD pixel time, i.e 22ns; the maximum error is always less than 44ns and the error is always pointing in the same direction so it can be corrected with half its value. Application of the averaging method will probably lead to a significant decrease of this error, however.

The geometric features of the CCD chip are excellent (close to ideal) thanks to the way it is produced, thus not producing any appreciable error. The geometrical behaviour of optics will not be ideal. However, these geometric imperfections are static. The AGAS can be calibrated once; the results of a measurement of a calibration DUT can be used in software to correct the non-ideal geometry of the optics and the CCD chip, thus deleting geometry errors.
For reasons mentioned before (VTG, CCD camera), the best way to measure a position on the screen is to determine the centre of a grey value pyramid that increases linearly from approximately 0% at the border to 100% at the centre of the dot. On each side of this pyramid some points with their grey values are detected. From this, a pyramid is reconstructed in software from which the centre can be calculated. The way the computer gets its information is by sampling the video signal coming from the CCD camera. This sampling, among other things, is accomplished by the PAPS.

The PAPS samples the incoming video signal on a rate $f_s$ (=15MHz). When only a passing of a certain grey level is detected, it can be calculated that the error due to the sampling will be $1/(2f_s)$. When, however, grey levels are taken into account the error will be reduced. If the grey levels could be represented by $\infty$ bits, the error caused by PAPS would be zero. But grey levels are presented in only 6 bits. The quantizing that is done for this reason, causes an error in the calculated centre of the pyramid. In the worst case only three 'grey value points' are used to reconstruct the grey value pyramid. It is assumed that two points (A and B) are taken from one side of the pyramid and a third point (C) is taken from the other side. Point C is assumed to be that point on the falling side of the pyramid that has a grey value which is closest to that of point A. A line is calculated through A and B; the steepness of this line is inverted to obtain the steepness of the line that is calculated through C. The point of intersection of the two lines is calculated as the centre of the pyramid.

The error made in this case is greatest when the quantizing error in B is maximal and the quantizing error in A and C is 0. This situation is shown in the next figure.

![Diagram of grey value pyramid and calculation of centre](image)

Figure 19. Worst case situation for the determination of the centre of the pyramid, using grey values.
The steepness of the raising side of real pyramid is called \( c_1 \), the steepness of the calculated line through A and B is called \( c_2 \). If the time difference between A and B is \( n \cdot T \) (\( T \) is the sampling period in PAPS, 66.7ns) and a unit step in grey value representation in PAPS (a change of the least significant bit) represents a change in grey value of \( \delta g \), then \( c_1 \) and \( c_2 \) are related as:

\[
c_2 = c_1 - \frac{\delta g}{n \cdot T} = c_1 - c_3 . \tag{V.1}
\]

It is known that the maximum difference in grey value between the calculated test figure and the real test figure, on the moment that the grey value of the real test figure is equal to that in point A, is \( \pm (c_1-c_2) \cdot T/2 \). Using this, the maximum error \( \delta t \) can be calculated:

\[
\delta t \leq \frac{(c_1-c_2) \cdot T/2}{2 \cdot c_2} = \frac{c_3 \cdot T}{4 \cdot c_1 - 4 \cdot c_3} . \tag{V.2}
\]

If, for instance, the pyramid is 2.0\( \mu \)s wide (30 PAPS samples) and 50 PAPS grey value units high and \( n = 1 \) (worst case), then the error is (\( \delta g = 1 \) PAPS grey value unit):

\[
\delta t = \frac{(1 \cdot T/T)}{(4 \cdot 50/1E-6 - 4 \cdot 1/T)} = 7.1 \text{ ns} .
\]

When on the falling side of the pyramid also two points (C and D) are taken, with grey values as close to those of A and B as possible, the steepness of the sides of the reconstructed pyramid can be calculated as the mean value of the steepness of the line through A and B and that of the line through C and D. This way the error is decreased considerably. When more than two points on each side of the pyramid are used, the error can be decreased even more. Depending on the definition of the test figure it will be possible to obtain a calculation of the centre of the pyramid in which the error due to the sampling of PAPS will always be less than less than 5.0 ns, from my point of view.

### V.2.7 Computer

Errors introduced by the computer will depend on the applied algorithms. Care has to be taken when calculations of very big or very small values are required. However I think that the computer won't introduce appreciable errors when programmed properly.

### V.2.8 Total error in horizontal direction

The errors made horizontally when using a CCD camera are estimated to be at most:

<table>
<thead>
<tr>
<th>Source</th>
<th>Error (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTG-20, VxE-20, EDS, DUT</td>
<td>15</td>
</tr>
<tr>
<td>Moire</td>
<td>25</td>
</tr>
<tr>
<td>Transition time CCD</td>
<td>22</td>
</tr>
<tr>
<td>PAPS</td>
<td>5</td>
</tr>
</tbody>
</table>

Total horizontal error: 67 ns
These times are times for the situation where the screen of the DUT is projected one on one on the CCD. When the projected screen width of the DUT on the CCD is \( p \) times the width of the CCD (\( p < 1 \)), the total error will be:

\[
\text{Total horizontal error} < 15\text{ns} + \frac{25\text{ns} + 22\text{ns} + 5\text{ns}}{p}.
\]

When \( p = 0.9 \) this is 73ns, which equals 0.15\% of the screen width of the DUT.

The mentioned numbers can decrease when a smart algorithm, using more points with their grey values as mentioned, is implemented.

### V.3 Errors made in vertical direction

The errors mentioned until now all refer to the horizontal direction. Vertically fewer errors are made. There will be no influence of moire because all vertical measurements will be done in or near the horizontal centre of the screen; on this place the screen surface is positioned parallel to the CCD so moire will only be in the horizontal direction. Transition times or bandwidths do not have any influence in vertical direction because of the way the DUT and the camera are functioning.

The vertical discreteness due to the line scanning in the DUT causes no error. The lines overlap so the picture on the screen will have a continuous character in the vertical direction. An error may be caused by the interlacing of both the DUT and the camera; it may lead to a vertical shift. To omit this, the test patterns have to be programmed non-interlaced and the camera is switched to non-interlaced. An advantage of this is that no attention has to be paid to the fact whether the picture taken is odd or even.

The only hardware influence will be caused by PAPS (the discreteness of the grey value representation). To calculate the error caused by this, a formula similar to (V.2) can be used:

\[
\delta h \leq \frac{(c_1-c_2)H}{2c_2} = \frac{c_3H}{(4c_1-4c_3)}.
\]

In this formula \( \delta h \) and \( H \) are given in percentage of the total screen height (%SH); \( H \) is the vertical sample pitch (line pitch = 100\%/275 lines = 0.36%SH). Assuming that the pyramid has a 'vertical width' (= physical height) of 4%SH (6 lines) and a height of 50 PAPS grey value units and \( n = 1 \), the error made in the vertical direction is:

\[
\delta h \leq \frac{1H}{H(4*50/2\text{SH} - 4*1/H)} = 0.011\%SH.
\]

Again, when the number of points that is considered in the calculations is increased, the error will decrease considerably. Additional errors can result from errors made in software, but will depend on the algorithm used. From my point of view, however, it can be stated that it must be possible to achieve the maximum error made vertically to be less than 0.10\% of the screen height of the DUT.
V.4 Conclusion

When the geometry of the DUT has to be adjusted within 0.8% of the screen width and 1.5% of the screen height, the AGAS will be accurate enough. Measurements in horizontal directions can be done with an error less than 0.15% of the screen width; vertically the error will be less than 0.10% of the screen height. When smart algorithms are developed together with the test pattern(s), these errors may even decrease.

When constraints will become more stringent more than one camera may be used, thus creating a situation where $p > 1$ (e.g. $p=>1.9$). This however may introduce difficulties like positioning requirements and optical deformation because the sensors are not 'looking' at the screen from the same point. It is not relevant at the moment, though.
VI SETTING UP THE AUTOMATED GEOMETRY ADJUSTMENT SYSTEM

To realise the AGAS a basic system is required in which specific solutions can be implemented. This basic system consists of two parts: hardware and software. The hardware is the physical part of the system, based on an IBM personal computer or compatible. With software the Architecture for Modular Test Systems is meant, which we use to create a user-friendly environment for making test programs.

The moment I needed such a system, it didn't exist. Only parts of it were available, so I had to build the system myself. In this chapter the way the system has been composed is discussed. First setting up the hardware will be dealt with, followed by a discussion of the software setup.

VI.1 Hardware

The hardware consists of a CCD camera, an IBM personal computer, an interface set called IGC-00, a VTG-20 MOBUS card, eventually one or more colour encoding cards, an EDS set, an IIC interface and PAPS cards called VAB, PTA and RTR. These parts have to be built together in one system as shown in the next picture.

![Diagram of hardware setup of the AGAS.](image)

Figure 20. Hardware setup of the AGAS.
In the shown setup the DUT is directly connected to the AGAS. In the production lines the AGAS will be implemented in the future, the DUT is mechanically fixed on a carrier in a defined way. This carrier is handled by the transport system. All signal and power connections to the outside world will be realized via sliding contacts under the carrier.

The IBM personal computer is a standard device, equipped with a keyboard, display device, (hard)disk drives, 2 parallel printer ports (Centronics), a serial port (RS232), real-time clock (battery backed-up) and free PC-bus connectors for extension purposes.

Both MOBUS and PAPS cards are controlled via the GPBS bus. To interface this bus to the computer the IGC-00 set is used, which consists of the IGC-00/S card and IGS-20 card. The IGC-00/S is plugged into one of the PC-bus connectors and mounted strongly. A flat cable, fitting in the IGC-00/S, leads to the IGS-20. This little card is connected to the GPBS bus at the back of a mounting rack.

Mounting racks are available for both MOBUS and PAPS cards. But because all cards have the same dimensions and only a few cards are used, it is preferable to combine PAPS and MOBUS cards in one mounting rack. Each rack contains two back panels, one of them representing the GPBS bus. The definition of the other one depends on whether it's PAPS or MOBUS. PAPS requires a PI(bus) in that place, while MOBUS needs the SCS bus. The PI bus contains 3 columns of connections; SCS only uses 2 columns. The two buses have to be electrically isolated from each other. The PI bus doesn't need any external connection; it is an internal bus of a PAPS.

A standard MOBUS mounting rack called GSP-26M, which consists of 26 places to fit cards in, is used to mount all cards in. The SCS bus plane is split up into two equal parts. One part is removed from the rack and a PAPS back panel, containing 13 places and configured as PI, is mounted instead. Such a back panel is a standard part in the PAPS product range. Now the rack has 13 places for PAPS and 13 places for MOBUS. The cards can be connected to the right bus by plugging them into the right side of the rack. MOBUS needs 4 places (VTG-20 plus 3 colour encoders); PAPS will take 4 places (VAU needs 2 places, PTB and RTR both need 1 place). Several places are left free, thus creating expansion facilities. Facilities are present on the back of the GPBS panel to connect the IGS-20 card.

Both MOBUS and PAPS cards use the power supplies implemented in the GPBS bus. Next to these, MOBUS cards need additional power supplies which have to be available on the SCS back panel. Industrial power supplying modules are mounted in a separate rack called MSR-20. A range of supplies can be composed using several individual modules; in this case +5V, -5V, +12V, -12V, +15V and -15V are created. The supplies required and the modules that are used have been summarized in appendix A.

To connect the power supplies to the other rack standard cables are used (SCS052, SCS053); the panels contain all required connecting facilities. A pair of ventilators in a frame is mounted between the racks to cool the electronics.

The VTG-20 and one or more colour encoders are interconnected via front end connectors on the cards; a flat cable is supplied with the cards for this purpose. The VAU has connections on the front
for four cameras and a monitor. Neither the PTB nor the RTR has any connection other than to the back panels.

The IIC interface is connected to the computer by plugging its cable into parallel printer port 2; printer port 1 is left free for the use of a printer. Connection of the interface to the TV set is done by using another cable that is supplied together with the interface; in the set that is used in experiments a service connector has been implemented for adding another device in the IIC structure. Both jumpers on the IIC interface are present, thus connecting pull up resistors to the SDA and the SCL lines. The dip switches are set to '1101' to obtain a clock frequency of 25kHz. The power supply (5V, 100mA) is derived from the 5V supply that is present in the MOBUS rack on the SCS bus (PS1).

For connecting the camera, a little connection board is built according to the following electrical scheme.

![Camera Connections Diagram](image)

**Figure 21. Electrical scheme of the camera connection board.**

Gamma correction in the camera is switched off, the scan mode is set to non-interlaced. The latter is done to become independent of whether an even or an odd frame is processed; otherwise a vertical shift may be caused by the interlacing. A requirement is that the test patterns are programmed to be non-interlaced, too. A potentiometer is implemented to adjust the gain of the camera; the gain control voltage is adjustable between 0V and 11V DC.

The external synchronization input and the video output are linked to coaxial connectors. The external sync input is connected to the composite sync output (X4) on the VTG-20, in order to omit that the camera will 'see' a black band coming over the screen due to a difference in synchronization frequency and/or phase. The video output is connected to camera channel 1 on the VAU. Connections are made using the appropriate cables. The camera is powered by the +12V logic supply on the GPBS bus. The EDS modules are mounted lying down inside the MSR-20 rack to save space; sufficient space is available to do so. A power supply of 24V/250mA is required. For this purpose the PE1114/02 (24V/2.9A) is used.
VI.2 Software

The software architecture called AMTS has to be applied in the AGAS. Software has been developed to build an AMTS that is working with the DRM operating system applied on an 68000 microprocessor. For the IBM personal computer and MSDOS not all software was available, yet. I partly had to produce this myself. To understand the creation of the AGAS software it is important to know how that software has been organized.

VI.2.1 Software organization

AMTS is built from several software bricks. One of the bricks is a library providing all general functionalities of AMTS, including an interface to the operating system. This AMTS library has already been converted to a library applicable on a personal computer, the $\mu$AMTS library. It doesn't contain anything dependent on any other hardware than a PC. This library has to be combined with other procedures, thus forming $\mu$AMTS.

In order to design a complete $\mu$AMTS the organization of AMTS for a 68000 system is considered. Based on that, an organization is composed for $\mu$AMTS.

VI.2.1.1 Existing AMTS

There are two software organizations in the 68000 version of AMTS, one for MOBUS and one for PAPS. Basically, they are the same; they only differ in the way the specific MOBUS/PAPS software is implemented. The way the software has been organised for MOBUS is depicted in the following figure; the position of the surrounding layers is shown as well. The top layer, labeled 'application', is referring to the application program that is implemented in the AMTS programming environment.

![Figure 22. MOBUS software organization in AMTS.](image)
The specific MOBUS software consists of only one layer containing all MOBUS card drivers. It is the interface between the application program running in AMTS and the hardware. Drivers have been developed for all MOBUS cards; every driver contains declarations of all functions applicable to the specific card. When building a MOBUS system the drivers are combined ('linked') with the AMTS library. In each driver a data brick has been defined according to a certain prescription, to interface all procedures to the AMTS programming environment. This way the test programming language (TPL) is extended with additional language features for controlling the cards.

MOBUS drivers have been written in C language. Interfacing to hardware is done using pointers to physical addresses; no operating system is required between the drivers and the cards. The operating system (DRM) only functions between AMTS library procedures and computer hardware. All communications between MOBUS cards and computer use AMTS library procedures and the operating system.

The software architecture applied with PAPS is shown in the next figure.

In PAPS a layer called HCR (Hardware Control Routines) interfaces all card service routines to the image processing hardware. In this layer all basic routines concerning input and output of control signals and data are gathered. All hardware interfacing is done using pointers.

A second layer in PAPS software contains two groups of general procedures called G(eneral) S(ervice) R(outines) and C(ard) C(ontrol) S(upport), and service routines for each of the PAPS cards have been implemented. The GSR and CCS routines are used by the card service routines. The GSR perform general functions like string analysis, adaption of input to a required format and error signalling. Therefore they have been interfaced directly to the operating system. The CCS routines take care for allocation of inputs and outputs of all PAPS cards on the PI bus, supply
facilities to change this and show it on screen. This is mainly done using procedures in the HCR layer; output to the screen is accomplished via AMTS library procedures.

In the MOBUS software organization every card has one driver module containing declarations of all functions applicable to that card; the interface to the programming environment has been included in this driver. In PAPS almost every function has been realised as a separate driver module (this applies to GSR and CCS too). The interface between the PAPS functions and the AMTS programming environment has been positioned centrally in a special source for this purpose, called OWNLIB. Not all functions have been interfaced this way, though; some are only serving others (e.g. GSR and CCS routines) so they don't have to be interfaced to the AMTS programming environment.

VI.2.1.2 μAMTS

As stated no AMTS is available for an IBM personal computer or compatible computer running on MSDOS, yet. Therefore an activity has been started to build such an Architecture for Modular Test Systems for micro computers, called μAMTS. This paragraph describes the μAMTS organization that is applied in the AGAS.

MOBUS and PAPS software have been organised differently, but they have a lot in common. Therefore I have tried to combine the organizations of both the MOBUS version and PAPS version of AMTS in a new organization, preserving the good features of each of them. This software organization of μAMTS is depicted in the following picture.

Both MOBUS and PAPS organizations are recognizable in the software structure of μAMTS. There are some differences compared to the AMTS organizations, though.

The hardware is split up into three parts, the IBM PC (or compatible computer), the MOBUS and PAPS hardware, and the IIC
card. The connection between software and MOBUS/PAPS hardware is made via the IGC-00 interface card.

Several control, input and output commands have been defined for the GPBS bus, as a part of the instruction set of P800M computers. The ones relevant to PAPS and MOBUS equipment are:

- WER(Address,Data) : output of Data to Address on the GPBS bus;
- RER(Address) : input of data from Address on the GPBS bus;
- OTR(Address,Data) : output of Data to Address on the GPBS bus (slightly different from WER!);
- INR(Address) : input of data from Address on the GPBS bus (slightly different from RER!);
- CIo(Address,Data,Mode) : Control commands for Input and Output, Mode indicates start or stop;
- TST(Address) : Test Status (before I/O operation);
- SST(Address) : Send Status (after I/O operation).

All control and data transfer in MOBUS and PAPS can be done using these commands. A driver is written that translates them to the right commands for the IGC-00, using standard C language I/O routines. This driver is called IGC CR (IGC-00 Routines). In MOBUS drivers for AMTS, input and output was done using pointers to the physical addresses; in PAPS, routines in HCR were called which performed the input and output in a similar way. In &AMTS, each pointer instruction to the GPBS bus is replaced by a relevant command from the list above. This will result in calls of IGC R, which will perform the control, input and output. So the original software organizations of MOBUS and PAPS are positioned on top of the IGC R.

For driving the IIC card new software is written. A routine called TRNSCV (supplied together with the IIC card by ELCOMA) is used for basic input from and output to the IIC card. A module on top of this routine contains functions for user-friendly operation of the IIC card from TPL.

The interface of PAPS procedures to the AMTS test programming environment was accomplished by a module called OWNLIB; in MOBUS it was a part of the driver. In &AMTS all required interfacing of MOBUS, PAPS and IIC routines is realised via OWNLIB. This way the software becomes well organised.

Some procedures are added to the AMTS library for making &AMTS more comfortable; facilities for entering DOS temporarily ('shell') and for performing direct input and output to any computer address (peek, poke, etc.) are created.

VI.2.2 How &AMTS is created

The executable file called &AMTS is composed from several software bricks. The grouping of all routines into bricks can be done in a lot of ways; here it is done analogous to the software organization of &AMTS.

All routines used to build &AMTS have been written in C or assembly language. These routines are compiled using the appropriate compiler. This results in a group of object modules that have to be linked together, thus producing an executable file.
In this case the number of object modules that has to be linked is very high (several hundreds of objects). There are a lot of routines required to obtain all basic functionalities of \(\mu\)AMTS; PAPS also uses a lot of routines. To simplify the linking process and to keep things clear, a library is composed from the objects of all routines that refer to a certain block in the \(\mu\)AMTS software organization; this is done for all PAPS blocks in the organization and for the \(\mu\)AMTS routines. This way seven libraries are created:

- \(\mu\)AMTS.LIB : containing all basic \(\mu\)AMTS routines;
- GSR.LIB : all General Support Routines;
- CCS.LIB : all Card Control Support routines;
- VAU.LIB : all routines for the Video Acquisition Unit;
- PTB.LIB : all routines for the Picture Transfer Buffer;
- RTR.LIB : all routines for the Real Time Recognizer;
- HCR.LIB : all Hardware Control Routines for PAPS;

When these libraries are used only six blocks in the \(\mu\)AMTS organization are not covered yet, and thus six object modules are left to be linked together with the libraries:

- OWNLIB.OBJ : interface to \(\mu\)AMTS programming environment;
- VTG.OBJ : routines for the Video Timing Generator;
- IIC.OBJ : routines for the IIC card;
- TRNSCV.OBJ : I/O routines for IIC.OBJ;
- IGC.R : I/O routines for PAPS and MOBUS;
- ADD.OBJ : additional routines

This way 13 files have to be linked together using the appropriate linker, to create \(\mu\)AMTS.EXE. When this executable is run the user ends up in the programming environment and can start implementing the application program.

VI.2.3 Collection of all software bricks

Most of the ingredients of the software bricks that are required to build \(\mu\)AMTS, have been written before for application in AMTS. Some of them haven't however, so they have to be written now. Newly written software concerns the IIC routines, IGC routines, the additional routines and the interface to TPL (OWNLIB, this is newly written because it is fully dependent on the used soft and hardware).

The software that already exists cannot be used immediately. It has been written for another computer running under another operating system, so the software first has to be adopted. In consultation with the writers of the original software it was concluded that this is possible without very big problems; therefore this software is ported.

The \(\mu\)AMTS routines (\(\mu\)AMTS library) have already been ported somewhere else, so they are available for building \(\mu\)AMTS. The version that is supplied to me is preliminary though; an optimised and more definite version will be available later.

The routines that are available to be ported to an IBM compatible computer running MSDOS are:
- all GSR routines;
- all CCS routines;
- all VAU routines;
- all PTB routines;
- all RTR routines;
- all HCR routines;
- the VTG module;

These routines are ported to the personal computer by me. Many changes that have to be made in many different places. But a systematic approach is possible because a lot of them have the same main cause. Therefore I won't explain every individual problem that is met in each source, but will mention all different problems, and the particular cause and the solution for each of them.

VI.2.3.1 Changes to port existing software

As said earlier, in AMTS all input and output was done using pointers straight to the physical address. This was a possibility in an 68000 system for fast I/O. Now these pointer instructions all are replaced by the appropriate input and output functions residing in IGCR (WER, RER, INR, OTR, CIO). The I/O functions used have to be imported to the actual module, by referring to the functions as 'extern' in the top of the source.

A feature of the compiler used for 68000 systems is that the type 'int' is 32 bits wide; a 'long' is also 32 bits and a 'short' means 16 bits. The Microsoft C compiler used for µAMTS handles an 'int' as being 16 bits wide. The term 'int' stands for 'integer' while 'long' means 'long integer' and 'short' means 'short integer'. In fact 'int' is machine dependent, while 'long' and 'short' are defined machine independent. The variables that are assigned as 'int' have to be 32 bits wide in this case. In order to achieve this and to create an unambiguous definition, all 'int' assignments are replaced by 'long'.

The µAMTS library routines have been defined in such a way that all integers that are used in a function call from TPL are put on stack as 'long's (32 bits). Therefore the C function has to take them off the stack as 'long's, too. Function declarations in C, where 'short's (16 bits) are used to transfer integer values, are altered to accept all integer values as 'long's.

A function that is altered this way may be called by another C function, too. Therefore in the calling function the integers it transfers have to be typecasted 'long'. This is taken care for in all sources. When variables in function declarations have been changed from 'short' to 'long', other variables may have to change as well.

When variables are printed on the screen using the C command 'printf( )', the formatting string in this command sometimes may have to be changed, too. In case the printed variable is assigned long, then in the format string '%d', '%i', '%lo', '%lu' or '%lx' has to be used instead of '%d', '%i', '%o', '%u' or '%x'. The latter may lead to the same result, but for instance when a long is put on stack and the latter is used in the formatting string, then from the moment the variable is printed the stack alignment is lost.

53
Some modules (C files) used by PAPS contain more than one function. In this case the order the functions are declared in the source is very important. At the time a function is called, that function must already be declared earlier in the source. This constraint didn't exist when AMTS was built for the 68000 system. For this reason, in some sources forward declaration of functions that are defined later in the same source, are added in the top.

The routine called TTRANB (PTB.LIB) checks the address of the place in memory where the first byte that is input from the Picture Transfer Buffer, is saved. In an 68000 system this address has to be even. In an IBM compatible computer this constraint doesn't have to be met anymore; the check is deleted. A module called VT100C (GSR.LIB) contains terminal related functions (highlighting, tab's, underline, etc.). Originally the module has been implemented for the use of a Digital VT100 terminal. The definitions applied in the module mainly are correct for the IBM personal computer. In the cases the definition are not right, they are modified.

A special source called PHWDAT.C needs some extra changes, because it is dependent on the configuration of the PAPS cards. In this source the configuration is defined, and for each PAPS card the addresses and default connections to the PI-bus are set in a predefined data brick. The information in PHWDAT.C is used during the initialization of the PAPS hardware. Additional to the standard PAPS modules some extra functions are ported creating an environment which is very useful in the experimental stage of the project. From this environment the PAPS cards can be operated interactively under operator control; the environment is called 'OPC'. This way an extra facility is added to the µAMTS programming environment which enables the programmer to try out things very quickly, without implementing them in TPL. In the OPC environment on-line help is permanently available.

VI.2.3.2 Self-written software

Next to the ported software, some modules still have to be written. As the names imply, all four modules are written in C language:

- IGCR.C
- IIC.C
- ADD.C
- OWNLIB.C

In IGCR.C all control, input and output commands that are relevant to the GPBS bus are defined:

- WER (address, data);
- RER (address);
- OTR_CIO (address, data, mode);
- INR_TST_SST (address, mode);

In the function calls for OTR_CIO and INR_TST_SST the mode determines what is meant: OTR0, OTR1, CIOHALT or CIOSTART, respectively INR0, INR1, TST or SST. Additional to these routines two extra routines are implemented; CLEARN (period) activates the clearn line on the GPBS bus during a period to reset all instru-
ments, INITGPBS (address) initializes the address of the IGC-00 card in software.

The IIC module contains two functions. The first one is called IIC_TRANSCEIVE and is used to perform all input and output to the IIC bus. Amongst others a parameter is present in the function call, that determines whether it is an input or an output operation that has to be done. This function also handles errors when they appear. The way the errors are handled is determined by another parameter in the function call; when an error is detected a text is printed on the screen or not, depending on this parameter. The error code is always available to the calling procedure when IIC_TRANSCEIVE has been executed.

Another function in IIC.C is called IIC_TRACE. This function is used to trace the IIC bus for a certain address and, when the address is detected, to store the data bytes sent to that address in an array for analysis. This feature is built in to be used in the future, when RC5 (remote control) commands in a t.v. set are acknowledged via the IIC bus by writing a certain code to a certain address. For the basic I/O to the IIC card an assembly routine called TRNSCV is used. This routine is delivered as an object module.

The module ADD.C contains functions for input and output of data in several formats to several locations. Next to these there is a routine called PTIME that prints the time on the screen, DSHELL lets you enter DOS temporarily and DOSDIR executes the 'dir' command.

OWNLIB.C is the module containing a data brick that is used to interface several C functions to TPL. To interface the functions they first are imported in the module. Then they are placed in the data brick in a defined way. As said not all C functions are interfaced to TPL; the functions that are interfaced have been summarized in appendix B. Nothing is said about the way the functions have to be called in TPL. In μAMTS an on-line help gives information about this; when this is not sufficient additional information is available at Philips CFT.

VI.2.4 Building μAMTS

At this point all required software bricks are available to build μAMTS; the basic μAMTS functions are available in μAMTS.LIB (preliminary version !!), the function TRNSCV has been supplied as an object and the rest of the functions have been written in C. In the creation of the executable file three stages can be distinguished. First all sources are compiled to obtain objects; several of these objects are gathered in a library. When the libraries and additional objects have been created they can be linked together to produce the executable μAMTS.EXE. To perform these steps the Microsoft C compiler version 4.0 is used. In contrast with its name MSC4.0 not only contains a compiler, but it also has a library utility and a link facility.

The MSC4.0 compiler uses memory models to gain control over how a program uses the memory of the IBM personal computer or compatible computer (e.g. stack-, heap- and code-allocation). This facility is applied because the PC memory is divided into segments of 64kbyte. By default the compiler uses the Small memory model; in that case the program code together with the data has to fit in one 64kb segment. In case of μAMTS the program
will be a lot larger, therefore the sources are compiled using
the Large memory model. The objects that have been gathered in
μAMTS.LIB have been compiled using the Large memory model, too.
Normally, when the Large memory model is used, both the stack and
all initialized global and static data are placed in the same
64kb segment. This is not acceptable in this case, because the
total amount of initialized data is so much that it doesn't leave
enough room for allocation of a sufficient stack. Therefore the
memory model is slightly changed to place the stack in another
64kb segment than the data. To apply such a memory model an
compiler option is used. For the normal Large memory model /AL
would be used, which is equivalent to /Alfd; now /Alfu is used.
But still not all data fits into one segment. Therefore another
option, setting a data threshold, is used: /Gt64. This option
causes all data items whose size is greater than or equal to 64
bytes to be allocated to a new data segment, thus decreasing the
amount of data that is allocated to the default data segment.
The MSC4.0 compiler has the possibility to use several alterna­
tives for performing mathematical operations. Differences between
these alternatives lies in their speed, accuracy and co-processor
support. Because no co-processor is available the sources are
compiled to use an alternate mathematic library to perform all
floating point operations; this is made clear to the compiler by
the /FPa option. This way the highest possible accuracy is

Using these three options the C-sources are compiled. A great
deal of the objects obtained are gathered in libraries. For
creating a library a program called LIB is used, which is
supplied together with the C compiler. As mentioned, libraries
are created for all functions related to the VAU, PTB, CCS,
GSR and HCR respectively.
The object modules of OWNLIB, IIC, TRNCSV, VTG, IGCR and ADD are
linked together with μAMTS.LIB, VAU.LIB, PTB.LIB, RTR.LIB,
CCS.LIB, GSR.LIB and HCR.LIB, using the linker supplied with the
MSC4.0 compiler. Additional to the libraries mentioned a library
called LLIBFA.LIB is linked. This is the alternate floating point
library mentioned earlier, especially created to use with the
Large memory model.
The linker by default allocates a stack of 2000 bytes. This
appears not to be enough; the link option '/St=10000' is used to
allocate 10000 bytes for stack. One of the Hardware Control
Routines used by PAPS is called 'rer'; it has been defined
slightly different from the function 'RER' I have defined in
IGCR. Because it is called by several PAPS routines, I'd like to
preserve this function. Therefore the link option '/NOI' is used,
which tells the linker to distinguish uppercase from lower case.
The number of software segments the linker is allowed to use
normally is 128, but because μAMTS is quite large I have to
change this. The maximum number of segments is set to 500 using
the '/SE:500' option.

When the linking process has finished the program named μAMTS is
ready to be used. Starting this program the user ends up in the
μAMTS environment where he can start programming in TPL or can
play around with PAPS using the OPC environment.
VI.2.5 Recommendations for future versions of μAMTS

The μAMTS just created is functioning very well. It may not be optimal though; some things may be changed depending on the requirements that have to be met in future developments. As said, the version of μAMTS just created contains the OPC environment; a lot of data is initialized by the OPC (help text). Later on, when the geometry alignment has been implemented in TPL, the OPC is not necessary anymore. Then all routines used by the OPC can be omitted in the building procedure. This may cause the amount of initialized global and static data to become less than 64kbyte. In that case the /Gt compiler option doesn't have to be used anymore, probably resulting in a faster program. It may even be possible to change /Alfu into /AL. The data threshold of 64 bytes that is applied during compiling, may be altered to a better value; this has not been investigated. It may also be advantageous not to apply the data threshold on every source, depending on how regular the particular source is used. This may lead to faster code, too.

Functions have been implemented in HCR for multiple I/O for PAPS cards. These functions are for instance used to input data from the PTB buffer. They are calling I/O routines in IGCR that perform the real I/O. To gain time the C routines from IGCR can be implemented in HCR immediately. It will save two function call procedures for every byte that is transferred. As said, the μAMTS.LIB that is used is not a preliminary version. This version will be optimized; it probably will become faster. What improvements exactly will be, cannot be said at this moment.

VI.3 Test and conclusions

Both software and hardware have been tested and are functioning well. The software has not been optimized for the final purpose, but for experimental purposes the realized system is very useful and user-friendly.

The interface from IBM personal computer to GPBS bus is functioning very well. The interfacing however could be improved by designing a similar board based on the IBM/AT extension bus instead of the IBM/XT extension bus. This will lead to a faster I/O (16 bits data instead of 8 bits), but will of course cost an amount of money. This improvement is not essential though.

The adjustment of the optics and the gain of the camera have not been considered. These will become actual when the picture acquisition problem is solved.
Now both hardware and software of the basic system have been completed, it is time to devote all attention to the development and implementation of a solution for the problem concerning automatic adjustment of the geometry of a t.v. set. The total development is split up into two parts, namely the development of control algorithms on one side and the acquisition of the pictures on the other. This chapter deals with the development of the control algorithms, which was preferred by my superiors to be done first. During the follow-up of the project, the acquisition of the pictures will be investigated and realised, for no time is left for me to do so. The implementation of the total solution in TPL will be done later, too.

VII.1 Starting points

Before the development of the control algorithms is started, a few basic things are assumed. These things are used as starting points in the development.

One starting point is defining the borderline between the control algorithms and the picture acquisition problem. The definition of test patterns will be related to the control algorithms. These test patterns will not be defined in much detail, though. Only things relevant to the algorithm will be defined, like 'a testing point is positioned on point \((x,y)\) in the test pattern' without saying whether this point will be a white dot, a box, a cross, or anything else. It is considered to be part of the picture acquisition problem to define this testing point and to retrieve the coordinates of the point in the picture taken by the camera, where the testing point appears. These coordinates are considered to be available for the algorithms, when applicable, within a sufficient range of accuracy. This assumption is made to free us from all kind of problems related to vision.

The test points are assumed to be very small compared to the size of the picture. It is probable that this is strived for, because otherwise extra problems are introduced due to the fact that the influence of a bad geometry on a test point will become considerable.

The horizontal position of the centre of the screen of the DUT is assumed to be known and the same for all possible DUT's. This is a fact according to the people operating in production plants. The t.v. set to be adjusted is assumed to contain a 3A type chassis as mentioned before. Sufficient statistics about sensitivities and values of adjustments are assumed to be available for this chassis.

No attention is paid to problems related to the size of the screen of the DUT, because this is a problem belonging to the picture acquisition part. Attention will be paid, however, to the way the size and position of the screen can be determined efficiently.

VII.2 The deflection circuit in a 3A t.v. set

A picture on a t.v. set is composed from nearly horizontal lines that are written by a scanning electron beam. The scanning is
caused by a magnetic field the beam goes through, which varies in time and is created by a deflection circuit. This circuit consists of a solid-state driving circuit and two deflection coils (one for the horizontal direction, the other for the vertical direction). The deflection coils are mounted on the back of the cathode ray tube (CRT), thus creating a magnetic field inside the CRT. This deflection circuit governs the geometry of the picture; it determines the way the scanning behaves as a function of time. The driving circuit has to produce such a wave form for each coil, that the electron beam in a CRT is deflected the right way all the time.

The horizontal driving signal has to behave monotonically during every line time, to cause the electron beam to scan from the left to the right of the screen. The vertical deflection driving signal has to be similar, but the period of this signal will be a lot longer, because during this period a lot of lines have to be written. The correct driving signals will not be linear, however, but will deviate from being linear, depending on the place on the screen the beam has to point at. This is caused by the way the CRT of a modern t.v. set has been shaped. The distance from a point on the screen to the electron gun is not the same for all points. Generally, when a coil is driven by a linear wave form, the deflection angle of the electron beam will vary linearly in time, resulting in a picture having the form of a deformed square (this is called 'pin-cushion distortion') and a very bad horizontal and vertical linearity.

Both a cross-section of the tube and the effects due to the construction of the tube, when the deflection coils are driven by linear signals are depicted in the following figure.

![PIN-CUSHION DISTORTION](image)

**Figure 25.** Cross-section of a CRT and its effect.

The coils and CRT used in 3A have been designed to correct the pin-cushion effect at the top and at the bottom of the screen (North-South correction) and the horizontal linearity, so these don't have to be adjusted anymore. Research at Philips has proved that correction of all other effects just meant can be accomplished by using a vertical deflection signal that behaves as a third order function of time, and an East-West deflection signal just containing zero, second and fourth order components. The vertical deflection signal controls the vertical component of the
scanning, the EW deflection signal takes care of the right horizontal scanning.

A circuit, typed TDA8432, has been designed that processes the signals required to obtain the right form of the picture on the screen. It can be implemented in any t.v. or monitor set that is based on the IIC bus. Amongst others, it has been applied as a slave processor in the 3A type chassis.

The algorithms for geometry alignment will partly be based on features of the TDA8432. Therefore it is necessary to know how this processor does the job. The end stages of the deflection circuit, in combination with the deflection coils are viewed as a black box from now on. The input signals it requires, and the effects on the screen when these signals are changed, are known however. Two driving signal are required, namely a vertical driving signal and an EW driving signal. The vertical driving signal is amplified to obtain the vertical deflection signal. The EW driving signal is inverted and then modulates a basic horizontal deflection signal which has a fixed amplitude.

The vertical driving signal will be periodic, with a period equal to the frame time in the set. Each period the signal has to go from a high voltage to a low voltage to let the scanning beam write its lines from top to bottom, and behave like a third order function to correct the non-linear behaviour of the vertical deflection coil and the tube.

The EW driving signal will have the same period as the vertical driving signal. This signal is to modulate the basic horizontal deflection signal as a function of time, in such a way that the pin-cushion effect will disappear and the picture width will be correct. It is known that in the end stage of the horizontal deflection circuit an inversion of the EW driving signal takes place. The horizontal deflection signal has to be decreased at the top and at the bottom of the screen to delete the pin-cushion effect. Therefore the EW driving signal has to have a maximum at the edges of the screen, resulting in a minimum in the horizontal driving signal at those points.

An example of what wave forms are thought of is given in the next picture.

**Vertical drive**

**EW drive**

**Figure 26. Example of possible deflection driving signals.**

In addition to the wave forms described, another feature related to geometry has been taken care of in the deflection processor. When the brightness of a picture is changed, it means that the electron beam current in the tube changes. Due to some properties of the end stage in combination with the tube (e.g. output resistance of a transformer in the end stage), the EHT (Extra High Tension, which is supplying the deflection coils), and thus
the amount of deflection, will change when the brightness of a picture changes. There is a direct relationship between brightness and the current in the scanning beam. Therefore, a correction has been implemented which measures the beam current and corrects the momentary driving signals dependent on this beam current in order to obtain a stable deflection.

All required wave forms are processed by the TDA8432 deflection processor by the use of an amount of registers, Digital to Analogue Converters, circuits performing arithmetical functions and some additional circuits. Both the vertical driving signal and the EW driving signal are derived from a sawtooth which is generated by the deflection processor internally. This is done because of the character of the required signals. The sawtooth has a fixed, frequency independent amplitude, which is ensured by an internal amplitude control loop; the DC offset voltage in the sawtooth has to be equal to its amplitude, so the waveform will vary between 0 and twice its amplitude.

A block diagram of the TDA8432 is shown below.

![Figure 27. Block diagram of the TDA8432.](image)

On the left in the block diagram, connections for the IIC lines (SCL and SDA) are situated; line A0 is used as the slave address and is activated by the master processor in the set, when applicable.

The vertical synchronization signal is input to the vertical flyback block; this block generates a vertical blanking signal with a pulse length that is determined by a resistor (pin 4, Iref is derived from an internal reference voltage) and a capacitor (pin 5). It also triggers the vertical scan block. This block generates the internal negative going sawtooth with a constant amplitude of 7.10V and a period determined by the vertical synchronization frequency. A direct current charges an external capacitor (pin 22) to obtain the linear behaviour of the sawtooth that is output to the geometry control block. This charging
current, which determines the amplitude of the sawtooth, is controlled by the amplitude control loop. A capacitor for the amplitude control loop is applied externally, because of its dependency on the field frequency of the set (pin 23).

In the IIC-bus interface all registers are residing, including the mechanism required to write to them. The contents of the registers are output to 13 DAC's. Three of them are not used for creating the EW-drive signal or the vertical-drive signal (DAC-A, DAC-B and DAC-C); they can be used for any purpose, e.g. for controlling the line frequency, the horizontal phase and a time constant of another chip in the set, the sync processor. The outputs of the 10 other DAC's are fed into the geometry control block; they function as parameters in the arithmetical creation of the driving signals. Another input on the geometry control block has been implemented for EHT compensation.

The input for a vertical feedback signal can be used to integrate a vertical correction mechanism in the TDA8432, depending on the way the deflection end stage has been realised. A possibility to control the polarity of the vertical drive signal has been implemented for the same purpose.

Facilities have been implemented for the use in manual or computer controlled alignment of the horizontal frequency in a set (pin 10 and 17); an IIC-controlled open collector output (pin 9) and a 3 level input that is interfaced to the IIC-bus (pin 11) were implemented for general purposes.

The part we are dealing with is considering the DAC's in combination with the geometry block. In the geometry block the basic vertical driving signal and EW driving signal are produced, using the sawtooth and the input signal for the EHT compensation as the start-off signals and the output of the DAC's as parameters.

For the vertical driving signal four geometry parameters are actual: Picture height, Vertical shift, Vertical linearity and Vertical S-correction. For the EW driving signal three parameters are relevant: Picture width, EW parabola and EW corner. All seven parameters have been implemented in the deflection processor as separate parameters. Next to these, there are two parameters for controlling the EHT compensation (Vertical compensation and Horizontal compensation), and one parameter called Trapezium. The latter has been implemented because of an unwanted DC offset in the internal sawtooth which will always be present; this offset is to be compensated by the output of the Trapezium DAC.

In this way ten parameters are available to the geometry control block, which all have been scaled in the processor to vary between 0 and 1. All parameters, including their related geometry parameters and the number of bits they are presented in in the deflection processor, are summarized in the next table.
Summary of all relevant TDA8432 parameters.

In order to simplify the equations given later for the driving signals, new variables are defined comprising the variables just mentioned in a simple way. These variables are:

\[- A = 0.85 \times \frac{(a+2.0)}{3.0} , \]
\[- D = 2.4 - 0.80 \times d \text{ Volt}, \]
\[- Y = 0.17 \times y , \]
\[- S = 0.44 \times s , \]
\[- W = 0.20 \times w , \]
\[- P = 0.60 \times p , \]
\[- C = 0.40 \times c , \]
\[- T = 0.32 \times (1-2.0 \times t) \text{ Volt}, \]

Furthermore, two extra variables are introduced:

\[ E = \frac{(V_{supp}/2-V_{EHT})}{40} , \text{ and} \]
\[ Z = (1-V_{saw}/3.55) + (V_{offs}+T)/3.55 . \]

\[ V_{supp} \] is the supply voltage of the processor, \( V_{EHT} \) is the voltage on the EHT compensation input of the chip, \( V_{saw} \) is the 7.10V sawtooth that is generated internally and \( V_{offs} \) is the DC offset voltage in this sawtooth. Variable \( E \) stands for the signal that is used to compensate for variations in the EHT in combination with \( v \) and \( h \). When no compensation is required \( E \) will be zero. \( Z \) is the basic wave form, from which the driving signals are derived. As mentioned Trapezium has to correct \( V_{offs} \); if it has been adjusted well, the second term in \( Z \) will be zero. In that case \( Z \) is a negative going sawtooth, varying between 1 and -1. All variables mentioned are applied in two formulas representing the vertical drive wave form and the EW drive wave form respectively:

\[ V_{vert} = D + 1.42 \times A \times (Z-S A^2 Z^3) \times (1-V_e) + Y Z^2 ] \text{ Volt} , \]
\[ V_{EW} = 30 \times [1-(1-W)(1-PA^2 Z^2+CA^4 Z^4)\times(1-hE)] + 1.2 \text{ Volt} . \]

As can be seen from these formulas, the driving signals are created by analogue multiplications and additions of the different signals and parameters. Almost all absolute factors that are used in the formulas, are directly derived from a stabilized internal reference voltage of
7.10V. The only factor that isn't, is the factor 30. This value actually is the value of the supply voltage. However it will not be used in any of the algorithms though.

VII.3 Algorithms for geometry adjustment of 3A sets

Based on the knowledge about how the deflection circuit in the 3A chassis is functioning, algorithms can be developed for adjusting the geometry of the picture produced on the screen of a 3A set. In this paragraph a solution is proposed for the total geometry alignment problem.

Next to the adjustment of the ten parameters mentioned, some extra things have to be arranged which will be mentioned in this paragraph as well. This concerns the measurement of the size and position of the screen and the positioning of the screen in software. The way the information about the screen is used to adjust the camera and/or optics, is not mentioned; this is considered to be part of the picture acquisition problem.

In the description of the algorithms, test patterns will be defined. These definitions sometimes will be illustrated by pictures of the patterns like they have to appear on the screen. In these pictures crosses will be used to point out where test points will be positioned. As mentioned earlier this doesn't say anything about the realization of the test points in the end.

VII.3.1 Order of alignment

In order to eliminate dependencies between different variables, a certain order is chosen to be followed in the adjustments. First of all the AGAS has to analyze what the size and position of the screen of the DUT is. This measurement doesn't have to be very accurate; it is only done to be able to adjust the equipment in a way that the screen is filling the picture taken by the camera for a great deal. When the AGAS has been adjusted to the right screen size and position, the screen has to be positioned in software to function as a reference for all other alignments.

The terms for compensating EHT variations in (VII.3), (l-vE) and (l-hE), will have unknown values. This might be an obstacle for the adjustment of the rest of the variables. To be able to do calculations as if there is no variation in the EHT, the compensation variables are adjusted first. When these have been adjusted well, calculations can be done as if there is no variation in the EHT and the terms (l-vE) and (l-hE) can be considered to be equal to 1, because then no correction has to be applied.

Before the vertical driving signal and the EW driving signal can be adjusted, the signal Z has to have the right form these signals are derived from it. This means the trapezium has to be adjusted next.

The vertical adjustments have to be done before the EW variables are aligned, because in the EW driving signal a variable, crucial to the vertical driving signal, is present (variable A); in the vertical driving signal no variables appears which are crucial in the adjustment of the EW driving signal. There are four variables applicable to the vertical driving signal (D, A, S, Y). with strong mutual influences on the screen. It will become evident that all four variables can be efficiently adjusted simultaneously.
The algorithm for adjustment of trapezium and the one to adjust all variables related to the vertical direction will be combined, because there is a strong relationship between these algorithms. When all this has been completed, four variables are left for adjustment. Three of them are related to the EW driving signal \((W, P, C)\), the fourth controls a separate DAC used for shifting the picture horizontally. In order to make it possible to adjust the first three variables efficiently, it is best to adjust the latter first because of the non spherical shape of the tube causing a non-linear behaviour. Nothing is gained by adjusting the three EW variables simultaneously for the same reason. They are adjusted sequentially.

The picture width operates on the total picture so it has to be adjusted first. It appears that the influence of corner correction is negligible (not noticeable) within a certain part of the screen. This fact is used to adjust the parabola correction, independent of the corner correction. When that has been done properly, the area on the screen where the corner correction is highly active, is known exactly thus guaranteeing the possibility of an efficient way of aligning the corner correction variable.

VII.3.2 Specification of the algorithms

In this paragraph the algorithms, considering the adjustments just mentioned, are specified as accurately as possible. Sometimes the settings on the instrument cards will be mentioned; not all settings will be mentioned in every case. Those that are not mentioned, will be in their default position. The default settings of all cards have been summarized in appendix C.

The size and position of the screen of the DUT have to be measured to adjust the picture acquisition equipment. This equipment has to cover a certain range of screen sizes. The size of the actual screen can be determined efficiently by first adjusting the picture acquisition equipment for the largest screen that can appear. The vertical centre line will always be in the same place; the aspect ratio of a screen will always approximate 1.33. The test pattern is chosen to be a plain white picture. Picture height and picture width are set at maximum \((a=1\) and \(w=0)\), which guarantees that the screen will be completely white. Because we don't need to know the size of the screen very accurately at this moment (this will be measured later; an inaccuracy of less than ±2% of the total height of the projected area is tolerated now), it suffices to measure either the height or the width of the screen. The horizontal position where the picture height is at a maximum, is known; this is the position of the vertical centre line. On that horizontal position a window is defined in the VAU with a minimum width (4 pixels) and maximum height (286 lines). Now the coordinates are wanted from only those pixels, having a relatively high grey value, because they point out the position of the screen. The PTB is programmed to store only the coordinates of these points (sparse packing in mode 4). From the collected y values, the screen height and the vertical position can be calculated. The horizontal position and aspect ratio are known, so the size and position of the screen can be calculated.
The relevant y values will cover a closed range; when values are found which do not fit to create a closed range, they have to be deleted because they will have been caused by noise. Because the accuracy constraints for this measurement are relatively low, it is sufficient to have information about only one pixel per y value. In order to reduce the number of processed pixels per y value to 1, a data rate restriction has to be set on the PTB to a value equal to 3, so that only the first pixel in a row containing 4 white pixels at most is processed. This will cause a negligible error, because the window is very narrow and the outline of the screen will be nearly straight and horizontal inside the window.

An extra data reduction is obtained by setting the vertical sub-sampling rate in the VAU at a value higher than 1. In my opinion it is acceptable to set a vertical sub-sampling rate of 5 (1 line is considered, then 4 lines aren't), causing an error of at most two times 4 pixels (worst case, sampled just before the first and just after the last white line). However the error made is always pointing in the same direction (calculated size will always be less than real size). Therefore the size calculated as the difference between the position of the two white borderline pixels is increased by 4 vertical pixel distances. This results in an error of ± 4 lines, which is less than ± 1.5% of the total height when only one frame is considered; the error caused by sub-sampling in calculating the centre of the screen will be less than 4 pixels, too.

In the following picture the idea of this measurement is illustrated for different screen sizes. The effect of setting a data rate restriction and the vertical sub-sampling is depicted as well.

Figure 28. Illustration of the determination of the size and position of a screen.

When the picture acquisition equipment has been adapted to the actual screen size and position (positioning, zooming), the screen has to be positioned in software. This means a reference screen is created in software, on which all other adjustments will be based.

It is assumed that the screen of the DUT will appear on the CCD sensor without any deformation due to the way it is projected; if the screen is flat and square, it is assumed to appear on the sensor as a square. This probably means that, in practice, the sensor is positioned parallel to the screen. When this is not the
case, the picture acquisition part has to imply a correction for it. Independent of whether the screen outline is rectangular or not, the picture on the screen has to be adjusted to be rectangular. The outline of the rectangle the adjusted picture is related to, is described by the horizontal and vertical tangent of the screen. The vertical position in the acquired picture where the screen is widest and the horizontal position where the screen is highest are measured; so are the width and height of the screen. These data can be used to setup a reference screen in software, related to the setting of the image acquisition part for the particular DUT. A possibility for measuring the outline of the actual screen is to use the RTR, detecting transitions from black to white or from white to black. This way the outline can be detected in one frame time. However, a feature of the RTR is that the horizontal resolution is halved, due to the time required to save an x-coordinate. Another disadvantage is that too much data is acquired. When the VAU and PTB are used for this purpose, the time required will be increased, but the accuracy will be kept at a maximum and the amount of data is minimized. Still using the white picture under the same conditions as before, while the environment is kept completely dark, the outline of the tube will be very clear. The PTB is set to store only the MSB of all incoming grey values (sparse packing in mode 2), thus detecting the pixel being white or black. The horizontal position of the screen centre is known. At this place a window is set with a maximum height (286 lines) and a width of 16 pixels (has to be a multiple of 16 because of the way of saving in PTB); this way less than 600 bytes containing 'grey value bits' and 300 y values will be input to the computer. The top, bottom and vertical centre of the screen can be calculated from the y coordinate of the first and the last grey value byte in the closed range of bytes unequal to 0, which describes the area the screen is positioned. Another window is set with a height of 6 lines and a maximum width which is a multiple of 16 (768 pixels); this window is centred on the coordinates that were calculated to be the approximate coordinates of the screen centre. The PTB mode is not altered. For each of the 6 y values approximately 100 bytes of grey value data are input plus one y coordinate. This data can be used to calculate the x coordinate of the left and the right border of the screen. From these values it is possible to calculate the x coordinate of the centre of the screen accurately.

The way the reference picture is determined, is illustrated in the following figure.
Figure 29. Determination of the reference picture.

The first deflection processor parameters to be adjusted are v and h, implemented for EHT compensation purposes. To measure the effect of the compensations effectively, two situations have to be compared: a situation where no compensation has to take place, and a situation where a lot of compensation is required. The variation in EHT is not equal for every place on the screen. This is due to a time constant which is delaying the influence of the brightness on the EHT every frame time. This time constant is caused by the internal resistance of the line transformer in combination with a capacitance due to the CRT. The order of the time constant is always several ms (e.g. $2\text{M} \Omega \times 2.5\text{nF} = 5\text{ ms}$ for a 3A set). The EHT is influenced by high brightness but the electron beam current, that is causing the brightness, first has to discharge the capacitance after the start of a new frame. Therefore, the variation of the EHT is greatest in the bottom of the screen while the picture contains a lot of brightness, because after several milliseconds (a frame lasts 20 ms) the capacitance has been discharged and the influence of brightness becomes considerable. When the picture is completely black, or has only a few little white spots in it, there’s no variation in EHT. When the picture contains a lot of brightness though, the EHT varies considerably.

In order to do the compensation adjustments, two test patterns are defined. One pattern, which does not cause any variation in EHT, is used as a reference in the adjustment. The other pattern is designed to cause a lot of variation in EHT. The test patterns are depicted in the following figure.

Figure 30. Test patterns for adjusting v and h.
In order to obtain a fairly good picture, all other geometry related parameters are set to values that are considered to be best, based on statistics. This is done to be quite sure that all test points are projected visibly on the screen. In practice the DUTs in a batch all need about the same adjustment values.

The test patterns both contain 4 test points. The two test points at the top and the bottom of the screen are used for adjusting v; h is adjusted using the points on the left and the right of the screen. The vertical test points are centred horizontally and positioned 5% of the screen height (5%SH) away from the top and bottom border. The horizontal test point positions are 5% of the screen width (5%SW) away from the left and right border respectively, and 10%SH away from the bottom border.

The test points will be fairly small, so no EHT variation will be caused when the rest of the picture is black. The window in the VAU is set at maximum (780 * 286) and the position on the screen of all test points from the first pattern are acquired. These positions are used as a reference. In the second pattern large white areas are implemented, causing a quick discharge of the CRT capacitance. The EHT will be different from the value it had when the first pattern was projected. The horizontal position of the vertical test points is known roughly, because the influence of the EHT variation on it is negligible on the vertical centre line and the horizontal positioning of the picture will be fairly good. A window is set, enclosing these two points. The position of the points is acquired, and the variation in distance, due to EHT variation, can be determined by comparing them to the original distance. An estimation is made for the required change in v, based on statistics about the sensitivity in earlier sets.

The result of this change is measured using the second pattern again. From now on a linear approximation, using the results of the two last measurements every time, is applied to adjust v. The same procedure will be used for adjusting h, except that the other two test points are considered.

The size of the windows to set and the white areas cannot be determined exactly now, because the size of the test points is not yet known. Another reason is that it is not exactly known how good the adjustments of the former set approach the required adjustments for the actual set. The better the adjustments, the more exact the windows can be set, and the larger the white areas can be.

The algorithm for aligning the vertical drive variables (D, A, S, Y) is based on the idea of an inverted function. A known vertical driving signal is programmed and the reaction on the screen is acquired. From the combination of the input signal (vertical drive) and the output signal (reaction on screen), the input signal which is necessary to get the required output on the screen, is calculated. A k axis is defined across the screen in the vertical direction; k=1 means the top of the screen, k=-1 means the bottom of the screen. When a position in the test pattern (video signal) is meant the term \( k_t \) is used (this is directly related to time); in case we're talking about a position on the screen of the DUT \( k_p \) is used. It is aimed to create a situation where \( k_t = k_p \) for all \( k_t, k_p \).

Four test points are defined in the test pattern, centred horizontally on \( k_{t1}=0.8, k_{t2}=0.4, k_{t3}=0.0 \) and \( k_{t4}=-0.8 \). All
geometry parameters that are not related to this algorithm and not have been adjusted yet (w, p, c, t) are set to the optimal values according the statistics of this type of DUT. Variables s and y are programmed to be 0. Variable A is given its nominal value (a=0.5, A0=0.71) and D is made D0=2.0 (d=0.5). This results in a linear vertical drive signal:

\[ V_{\text{vert}} = D_0 + 1.42* A_0 * Z * (1-\text{vE}) = 2.0 + Z \text{ Volt.} \]  

(VII.4)

The test pattern and the vertical drive signal are depicted in the following figure. A relation is shown between the vertical drive signal and \( k_t \).

![Figure 31. Test pattern and vertical drive signal for alignment of the vertical drive parameters.](image)

In this case the vertical drive signal is a linear signal. But due to T, which has not been adjusted yet, there may be an offset in Z. When T has been adjusted well this offset will be 0 and the value of Z for \( k=0 \) will be equal to 0. The value of \( V_{\text{vert}} \) for \( k_t=0 \) is called \( V_{\text{vert},0} \). The steepness of the linear vertical drive signal is called \( q_0 \). Because \( q_0 \) is dependent on characteristics of the circuit and applied components, I don't like to calculate it from general data about the chip. By acquiring an extra picture (with changed settings) the steepness \( q_0 \) is measured. The value of \( V_{\text{vert},0} \) is measured as well, using a third picture. Using these variables, the vertical drive signal is described by:

\[ V_{\text{vert}} = V_{\text{vert},0} + q_0 * k_t \]  

(VII.5)

When both \( V_{\text{vert},0} \) and \( q_0 \) are known, the vertical drive voltage is known for the places where the four test points reside: \( V_{\text{vert},1} \), \( V_{\text{vert},2} \), \( V_{\text{vert},3} \) and \( V_{\text{vert},4} \). The picture, resulting from this drive signal and the test pattern shown, is acquired and the real positions of the test points on the screen, are measured: \( k_{p1}, k_{p2}, k_{p3} \) and \( k_{p4} \). Now it is known that \( V_{\text{vert},i} \) is required to result in the right vertical position \( k_{pi} \). Because we have 4 points, a third order function can be calculated, which describes \( V_{\text{vert}} \) as a function of \( k_p \) (or \( k_t \), because when the right drive signal is applied then \( k_p=k_t \)), required to obtain a linear transformation from the time domain (test pattern) to the vertical position domain on the screen. This function is calculated by solving four equations:
\[ V_{vert}(k_p) = a_0 + a_1 k_p + a_2 k_p^2 + a_3 k_p^3 = V_{vert,0} + q_0 * k_t \]
\[ V_{vert}(k_{p2}) = a_0 + a_1 k_{p2} + a_2 k_{p2}^2 + a_3 k_{p2}^3 = V_{vert,0} + q_0 * k_t \]
\[ V_{vert}(k_{p3}) = a_0 + a_1 k_{p3} + a_2 k_{p3}^2 + a_3 k_{p3}^3 = V_{vert,0} + q_0 * k_t \]
\[ V_{vert}(k_{p4}) = a_0 + a_1 k_{p4} + a_2 k_{p4}^2 + a_3 k_{p4}^3 = V_{vert,0} + q_0 * k_t \]

These equations deliver \( a_0, a_1, a_2 \) and \( a_3 \), which altogether define the required vertical drive signal \((k=k_p=k_t)\):

\[ V_{vert}(k) = a_0 + a_1 k + a_2 k^2 + a_3 k^3 \]  \hspace{1cm} (VII.6)

and because \( q_0 * k_t = 1.42 * A_0 * Z \) when \( T \) has been adjusted ideally:

\[ V_{vert}(Z) = a_0 + a_2 * Z^2 + a_3 * Z^3 \]  \hspace{1cm} (VII.7)

From this equation combined with equation VII.3, the required values for \( D, A, Y \) and \( S \) can be derived:

\[ D = a_0 \]
\[ A = A_0 * a_1 / q_0 \]
\[ Y = (1.42 * A_0 * a_2) / (q_0 * a_1) \]  \hspace{1cm} (VII.9)
\[ S = - (1.42^2 * a_3) / a_1^3 \]

If \( T \) has not been adjusted ideally, the next equation is valid:

\[ q_0 * k = 1.42 * A_0 * Z + D_0 - V_{vert,0} \]

or

\[ k = \frac{1.42 * A_0 * Z}{q_0} + \frac{D_0 - V_{vert,0}}{q_0} = k_0 + \delta k \]  \hspace{1cm} (VII.10)

Here \( k_0 \) is the value of \( k \) for which \( V_{vert} = D_0 \). Equation (VII.10) is true for every \( T \) ! Substituting this in (VII.7) leads to the following equation:

\[ V_{vert}(k_0) = (a_0 + a_1 * \delta k + a_2 * \delta k^2 + a_3 * \delta k^3) + (a_1 + 2 * a_2 * \delta k + 3 * a_3 * \delta k^2) * k_0 \]

\[ + (a_2 + 3 * a_3 * \delta k) * k_0^2 + a_3 * k_0^3 \]

So \( V_{vert,0}(Z) = (a_0 + a_1 * \delta k + a_2 * \delta k^2 + a_3 * \delta k^3) + \frac{(a_1 + 2 * a_2 * \delta k + 3 * a_3 * \delta k^2) * (1.42 * A_0 * Z / q_0)}{1.42 ^ 2 * A_0 * Z / q_0} + \frac{(a_2 + 3 * a_3 * \delta k) * (1.42 * A_0 * Z / q_0)^2}{1.42^3 * A_0^3 * A_3} + a_3 * (1.42 * A_0 * Z / q_0)^3 \]

When (VII.8) and (VII.11) are compared it is obvious that the following settings have to be realised:
\[ D = b_0, \]
\[ A = A_0 * b_1 / q_0, \]
\[ Y = (1.42 * A_0 * b_2) / (q_0 * b_1), \]
\[ S = - (1.42^2 * b_3) / b_1^3, \]
with \( b_0 = a_0 + a_1 \delta k + a_2 \delta k^2 + a_3 \delta k^3 \),
\[ b_1 = a_1 + 2a_2 \delta k + 3a_3 \delta k^2, \]
\[ b_2 = a_2 + 3a_3 \delta k \quad \text{and} \quad b_3 = a_3. \]

But we first adjust \( T \) because this has to be done optimally before any of the other adjustments is considered. \( T \) is adjusted using the measured \( V_{\text{vert,0}} \). When an offset \( \delta Z \) is present in \( Z \), this means that the value of \( V_{\text{vert}} \) at \( k_t=0 \) is increased from \( D_0 \) to \( (D_0 + 1.42 * A_0 * \delta Z) \). The value of \( D_0 \) is known, so when \( V_{\text{vert,0}} \) has been measured \( \delta Z \) can be calculated. According to equation VII.2 the change in \( T \) to correct this \( \delta Z \) is:

\[ \delta T = T' - T = -3.55 * \delta Z, \]

so the new value of \( T \) (\( = T' \)) has to be:

\[ T' = T - 3.55 * \frac{V_{\text{vert,0}} - D_0}{1.42 * A_0} = T - 2.50 * \frac{V_{\text{vert,0}} - D_0}{A_0}. \]

According to people dealing with production the required value for \( T \) is equal for almost all set of a particular type, in practice. This means it is imaginable that in many cases no correction of \( T \) is required. However, \( T \) will not always correct the offset in the deflection processor for 100% because it is a digital setting.

As said, for the adjustment of \( T, A, D, S \) and \( Y \) the value of \( q_0 \) and \( V_{\text{vert,0}} \) has to be known. One picture is acquired for obtaining information \( (k_{pi}) \) to calculate the values of \( a_i \). The important parameters related to this picture are \( D_0=2.0, A_0=0.71, S=0 \) and \( Y=0 \). Now \( D \) is changed to \( D_1=2.2 \) and an extra picture is acquired (still using the same test pattern) to calculate \( q_0 \). The test point at \( k_{t4} \) is used in this measurement, because this point is positioned most isolated from the other points. It is known on what \( k_p \) value it is projected when \( D=D_0 \) from the former picture \( (k_{p4}) \). The new picture is acquired and the \( k \) coordinate of the place it is projected now, is available \( (k_{p4a}) \). From the data that was already available after the former measurement, a relationship between \( k_t \) and \( k_p \) in the form of a third order function can be calculated for \( D=D_0, A=A_0, S=0 \) and \( Y=0 \):

\[ k_t = c_0 + c_1 * k_p + c_2 * k_p^2 + c_3 * k_p^3. \]

This calculation is based on 4 couples \( (k_{pi}, k_{ti}) \). By substituting \( k_{p4a} \) in this equation, \( k_{t4a} \) can be calculated. This is the value of \( k \) in the former equation of \( V_{\text{vert}} \) \( (D=D_0) \), for which the electron beam was projected on \( k_{p4a} \).
From all information gained $q_0$ can be calculated:

\[
V_{\text{vert},4} = D_0 + q_0 k_t4
\]
\[
V_{\text{vert},4a} = D_0 + q_0 k_t4a
\]
\[
V_{\text{vert},4a} - V_{\text{vert},4} = D_1 - D_0
\]

The calculation of $q_0$ is illustrated in the following figure.

![Figure 32. Calculation of $q_0$.](image)

When the steepness $q$ of the linear vertical drive signal is altered, the wave form that describes this drive signal will rotate around a certain point $R$ in the time-voltage plane: $R = (k_R, V_R)$. The voltage $V_R$ on this rotation point will always be equal to $D_0$, independent of the fact whether $T$ has been adjusted well or not. $T$ doesn't influence the steepness either; a change in $T$ only shifts the rotation point in time ($k_R$). If $T$ would have been adjusted ideally then $R$ would be $(D_0,0)$. $V_{\text{vert},0}$ can be determined by altering the value of $A$ from $A_0$ to $A_1=0.64$ ($a=0.25$); $D$ is kept equal to $D_0$. Again the test point at $k_t4$ is used. A picture is acquired when $A=A_1$; the $k$ coordinate where the test point appears now is $k_{p4b}$. This value is input to the equation VII.15 to calculate $k_t4b$. An illustration of the situation is given in the next picture.
The distance between \( k_{t4b} \) and \( k_R \) is called \( \delta k_b \); \( \delta k \) is the distance between \( k_{t4} \) and \( k_R \). It is obvious that:

\[
\frac{\delta k_b}{\delta k} = \frac{A_1}{A_0}, \tag{VII.17}
\]

or

\[
\frac{A_1}{A_0} \times \delta k = \delta k_b = \delta k - (k_{t4b} - k_{t4}) \tag{VII.18}
\]

implying:

\[
\delta k = \frac{A_0}{A_0 - A_1} \times (k_{t4b} - k_{t4}) \tag{VII.19}
\]

so

\[
V_{\text{vert}, 4} = D_0 + q_0 \times \frac{A_0}{A_0 - A_1} \times (k_{t4b} - k_{t4}) \tag{VII.20}
\]

But it is also known that:

\[
V_{\text{vert}, 4} = V_{\text{vert}, 0} + q_0 \times k_{t4} \tag{VII.21}
\]

and therefore

\[
V_{\text{vert}, 0} = D_0 + q_0 \times \frac{A_0}{A_0 - A_1} \times (k_{t4b} - k_{t4}) - q_0 \times k_{t4} \tag{VII.22}
\]
When three pictures are processed as described and $V_{\text{vert},0}$ and $q_0$ are known, equation (VII.7) can be solved. Variable $T$ can be adjusted using equation (VII.14). This will result in a partial but optimal correction of the offset in the deflection processor. The value $(D_0 - V_{\text{vert},0})$, which results from the part of the offset that couldn't be corrected by $T$ (so here $V_{\text{vert},0}$ means the value of $V_{\text{vert}}$ on $k = 0$ after $T$ has been adjusted), can be calculated. This can be used together with the other knowledge gained, to adjust $A$, $D$, $S$ and $Y$ by applying equation (VII.12). This way the vertical adjustments are accomplished with the information from only 3 pictures.

When the vertical drive parameters and trapezium have been adjusted, the only parameters left are those applicable to the horizontal direction: DAC-B for horizontal shift, $W$, $P$ and $S$. For alignment of all four parameters one test pattern is defined as depicted below. It contains one test point in the centre of the pattern and three other points positioned $5\%SW$ away from the left border, respectively $50\%SH$, $40\%SH$ and $5\%SH$ away from the top border.

![Figure 34. Test pattern used to adjust DAC-B, W, P and C.](image)

All four parameters are aligned separately; combination of different adjustments is not advantageous because the separate adjustments are very simple.

A basic algorithm is used for all four parameters. Initially the parameter is set on the right value of the former set that was adjusted. Most of the time, this will approximate the required value very well. A window, enclosing the relevant test point only, is set and the response on the screen is measured efficiently. If the adjustment is not optimal, an estimation is made for the value that is required, based on statistics about the sensitivity of the particular adjustment. From that moment on, a linear approximation process is started to obtain the right adjustment. This linear approximation will converge within only a few cycles; because the initial setting probably is very good and because of the fact that therefore the interesting area will be very small, the sensitivity of the adjustment within that area will approximate a linear function.

Test point 1 is used to align horizontal shift, point 2 is implemented for picture width, point 3 is for parabola correction and point 4 is functioning as the test point for the corner correction. This particular position of point 3 is chosen because in sets using the TDA8432, there is an area on the screen in which the influence of corner correction is negligible; this area.
has a height of 30%SH and is positioned along the total width of the screen, symmetrically around the horizontal centre line. This makes it possible to adjust the parabola and corner correction independently. The position of point 4 is chosen at an extreme position because the effect of corner correction is at a maximum in the utmost corners of the screen.

There are some facts that are relevant to all algorithms specified. They will especially become important in the implementation of the algorithms.

First of all the discreteness of the variables has to be taken into account in the decision whether the setting is optimal or not. The setting may not be perfect, but it may be the one closest to the perfect setting, which cannot be realized due to the binary presentation that is used.

The statistics that are sometimes required don't have to be available at this moment. They can be gathered during the first adjustments, too. An alternative is to use the values of the former set that has been adjusted.

When D is altered, the t.v. set reacts with an overshoot in the position of the picture on the screen. Every time D is changed one has to wait for 150ms before acquiring the next picture to eliminate any influence from this feature, according to the designers of the set.

Another point to bear in mind is a possibility for optimization. It may be advantageous to combine the end of one adjustment with the start of the other one. For instance while checking the adjustment of W, the coordinates of the test point for P can be obtained from the same picture by setting the right window. This saves the acquisition of one picture, every time the former adjustment has been right. This method however, costs some extra time to calculate the extra information every time; it has to be determined experimentally whether it is advantageous or not.

The setting of brightness and contrast in the set is part of the vision problem. It may however be important to the control algorithms too, especially to the one in which the EHT compensation variables are aligned. Not much can be said about this now; experiments may lead to an optimal solution.

The memory on the VTG to store patterns in, is limited. However, looking at patterns programmed until this moment, I think it will suffice to program all required patterns.

VII.3.3 Time and accuracy considerations

To get an idea how time is consumed or available, an overview is given about how much data is acquired and how many time is needed for this. In this investigation a possible overlap of algorithms as proposed above is not taken into account.

The mean amount of data to be transferred from PAPS to the computer after the acquisition and processing in hardware of one picture is assumed to be less than 750 bytes; this number is based on experience and the test patterns that have been defined. The transfer of one byte takes about 175 μs on a P3200 computer running on a 6MHz clock frequency. This means that the average time taken by the transfer of data from one picture will be less than 130ms. The acquisition and hardware processing of the picture will cost one frame time, which is 20ms. On the average
the PAPS has to wait 10ms before the start of the next frame; this has to be taken into account as well. So summarizing, it can be assumed that the mean time required for getting relevant data available in computer memory will be less than 160ms per picture. How many pictures will have to be processed? The initial determination of the size and the position of the screen takes 1 picture. For calculating the reference picture 2 pictures are required. The first step in the adjustment of v and h is done using 1 picture. After that, from my point of view it will be possible to adjust v and h in two times two steps, which means that two times 3 pictures are acquired. The alignment of T, A, D, S and Y requires only 3 pictures. The adjustment of DAC-B, W, P and C each may need four pictures at most, I think. Altogether it seems that 29 pictures have to be acquired to adjust all geometry parameters. This means that less than 5 seconds are spent in the acquisition of the pictures. Time required to solve vision problems are not included. When 25 seconds are available, 20 seconds are left for vision algorithms, calculations in the control algorithms and control of other instruments. The control of the instruments won't take much time, because this implies the transfer of a few control bytes which only has to be done a few times. I think it is reasonable to say that at least 18 seconds will be available for software processing. In TPL an operation on reals (addition, substraction, multiplication, division) takes less than 0.70 ms on a 6MHz IBM/AT compatible. An access of a one dimensional array takes about 0.18 ms; accessing a two dimensional array lasts 0.36 ms. The time required for array access is independent of the size of the array or the place that is accessed. Assuming that the number of reals operations and array accesses is equal, this means that about 20000 operations and 20000 accesses can be done in the time available. I estimate that the number of operations and accesses will be at least 5 times less. Taking all this into account the available time seems sufficient to me.

The original accuracy requirements were based on the assumption that four positions at most had to be compared. This appeared not to be reality. The errors made in the adjustment of the horizontal position of the picture consists of the error made in the creation of the reference picture in software and the error made in the measurement of the actual position of the picture on the screen. The first error will always be less than about 100ns (CCD camera and PAPS sampling), which means \( p = 0.9 \), see V.2.9) 0.20%SW; the second error will be less than 0.15%SW (see V.4). So the total error will be less than 0.35%SW and the margin in software for adjustment of the horizontal position is 0.90%SW (± 0.45%SW). The picture width is adjusted in relation to the measured screen width and the centre of the picture. Again the error in the measured screen width is less than 0.20%SW, the error in the horizontal position of the screen will be less than 0.35%SW. Totally, the error made in this case will be less than 0.55%SW, leaving a 0.50%SW (± 0.25%SW) margin for adjustment. Both parabola and corner correction measurement are referred to the position of the test point used to adjust the picture width. Therefore the maximum error made in both cases is twice the error made in the measurement of a position of a test point, i.e. \( 2 \times 0.15\%SW = 0.30\%SW ; 0.50\%SW \) is left for adjustment of these
parameters in combination with trapezium. In the measurements to adjust the EHT compensations, two positions are compared. This means the maximum error made horizontally will be less than 0.30%SW; vertically the error will be less than 0.20%SH. The vertical margin for adjustment therefore is 1.3%SH, the horizontal margin is 0.50%SW.

It takes some calculation to see what the maximum error will be in the adjustment of the vertical geometry parameters. But without calculating anything, sensible things can be said about the accuracy of these adjustments. The requirements are not very severe; A and D have to be adjusted with an accuracy of 1.5%SH, S and Y have to produce an accuracy in the picture of 2.0%SH. The measurements in vertical direction can be done very accurately (0.10%SH). Due to the method followed, the errors caused never will exceed a few times the measurement errors. Therefore I think it is very well possible to adjust the parameters within the required accuracy. Trapezium will be adjusted optimally, so a great deal of the 0.50%SW margin will be left for the adjustment of P and C; I assume that at least 0.30%SW will be available.

The available margins all suffice. When the step in the picture caused by a step in any parameter is calculated, assuming that a right adjustment implies that all variables are set at half their maximum value, it appears that they are staying within the margins (δa=1 is equivalent to a change of 0.6%SH, δd => 0.6%SH, δw => 0.35%SW, δp => 0.12%SW, δc => 0.16%SW).

VII.4 Conclusions

Algorithms have been specified for all variables in a 3A set that are related to geometry, including the test patterns required. These specifications are not always very accurate. Sometimes unknowns are left that have to be filled in, based on experiments or dependent on, for instance, the vision part. But in spite of the incomplete specification, I think the adjustments are done optimally. The test patterns have been defined to minimize the influence of non-relevant variables on relevant variables in each algorithm.

The accuracy of the measurements that are done in the algorithms leaves sufficient room for the digital adjustments. The time left for software processing is sufficient I think. The way the algorithms have been specified will be appropriate to implement them in TPL in a straightforward way.
VIII RECOMMENDATIONS FOR FOLLOW-UP

The development of an automated AGAS has not been finished yet. Only part of the problems have been solved. A major problem that is still to be solved is the picture acquisition. Related to this is the definition of the test points in the test patterns that still has to be done. The variation in screen sizes has not been considered either.

In spite of the fact that these things have not been solved yet, some ideas came up during the time that was spent on other subjects. Some of them were already mentioned in this report. In this chapter some tips are gathered which may be very useful for the follow-up of the project.

Concerning the picture acquisition some things were said in chapter VII when the control algorithms were described. It appeared to be possible to align all geometry parameters efficiently without using the Real Time Recognizer. Windows are set and data is stored in certain ways to reduce the amount of data that has to be transferred to the computer and evaluated. The camera is set in the non-interlaced mode to omit dependency on whether the picture is odd or even; the test patterns therefore are programmed to be non-interlaced, too.

It was assumed that the horizontal position of the centre of the screen is known from the start. Starting at this point the vertical position and the size of the screen are determined. In the future flow line however an active identifier is detached to the carrier the DUT is standing on. This identifier contains information which it supplies to for instance the transport system. But this information can be used by the AGAS as well. A data base can be setup containing the screen size and position relative to a reference point for all possible t.v. sets in a factory. From the identifier the set type can be read and the AGAS can be adjusted immediately. The reference picture in software can be defined immediately, too. However, this demands a very high accuracy of the mechanism that is used to adapt to the particular screen size; it may be advantageous (cheaper) to do it every time again, using the camera. The data base may also contain information about nominal settings of each set.

The mechanism to adapt to a screen size can be realized in several ways. Because the VAU has four camera inputs it is possible to use four cameras, each of them positioned in a specific place and/or utilized with specific optics. Another possibility is to use a mechanical installation to move one camera up, down, back and forth. This installation could be controlled using a MOBUS motor control card. A mixture of these two possibilities, the use of several cameras which can only be moved in one direction, is an alternative, too. This deletes the disadvantage of the first method that not all picture sizes can be inspected optimally. Another possibility to delete that disadvantage is to use even more than four cameras, which all have been adjusted to a particular screen size, and switch the signals to the VAU using a MOBUS card containing coaxial switches.

The camera used uses an D/A conversion in the time domain to obtain an analogue output signal. In PAPS the signal, which behaves analogue in time and value, is sampled and quantized by an A/D converter thus producing digital data. It may be ad-
vantageous to delete the two conversions and only implement an A/D converter for digitizing the momentary analogue values produced by the CCD. The 'sample frequency' in the camera (11.55MHz) is less than the sample time in PAPS (15MHz), but this may be compensated by the decrease of other errors that were caused by the conversion just described.

The test points have to be designed as simple as possible to minimize the amount of irrelevant data that has to separated from the relevant data. To decrease several errors in the AGAS it already has been proposed in chapter V to apply pyramids of grey values in a test figure, then reconstruct these pyramids from the results on the screen and calculate the mean value of the centre of these pyramids, which is the point of interest. This idea obviously has to be applied only in the relevant direction. Horizontal test points have to be pyramidal in horizontal direction only; in vertical direction the points have to have a grey value square wave, thus containing as less information as possible. Data about more than one line in such test point can be used for averaging to decrease the error made. The same idea is applicable on vertical test points.

The adjustment of geometry has to be independent of as many other adjustments as possible. One adjustment in particular is of interest here: convergence. The convergence adjustment makes sure that the beams of the blue, green and red electron gun are always pointing at the same point on the screen. When this adjustment has not been done well yet, it can lead to an extra error in the measurements for geometry adjustments. To omit this, the test pattern can be programmed to contain only green, because the green gun is positioned in the centre of an in-line CRT and the beam will not be affected by the convergence adjustment. This however has a disadvantage.

When only green is used in the test pattern, the moire effect is at its greatest. This can be a problem. A solution to this is not to focus on the screen. The moire effect will be completely deleted this way, but now another problem is introduced: the picture will be deformed. However, I think it will be possible to describe the deformation mathematically. When that's possible an inverse transformation can be applied to obtain the original geometrical information. A possibility to employ the transformation may be the application of a PAPS hardware processor (e.g. WLT); it may be accomplished in software as well. This way the moire effect might be deleted. It may also be possible to delete moire when the camera is focussed on the screen, using one of the PAPS hardware processors. Investigations and experiments have to prove what solution is preferable.

The optics will not be ideal geometrically. The AGAS will have to be calibrated to correct this fact. This correction may be combined with the transformation function that was used in deleting moire. To minimize the geometrical deformation caused by the lenses (fish eye deformation), the diaphragm has to be kept as small as possible. The gain of the camera can be adjusted to obtain sufficient output. It has not been defined what light output or input is called white or black. Neither any attention has been paid to the adjustment of contrast and brightness. These quantities have to be determined experimentally.
An AGAS has been defined meeting all requirements. Several CFT concepts are applied adjacently in this system, which is rather unique. It has been tested and is functioning well as expected. The µAMTS which has been created for the IBM personal computer is not the definite version. The software brick which is required for this will become available in a few weeks time. The software has been configured clearly and structured enough though, to allow other people to compose a new µAMTS when possible. The actual version comprises some extra features which are very useful in the experimental stage.

Generally applicable constraints for geometry alignments have been defined. Control algorithms, meeting the accuracy requirements, have been developed based on the IIC deflection processor TDA8432. A part of these algorithms has been designed especially for this processor. In the near future (several years) this will be the main component used in deflection circuits of Philips monitors and t.v. sets, so this isn't a problem. When it isn't used anymore this part of the algorithms has to be changed, though. However, things can be learned from the TDA8432 algorithms. It shows an approach which is completely different from conventional methods, but is very efficient.

The algorithms have not been entirely worked out; it is for instance not shown how equations have to be solved or a function going through some points can be calculated. It is assumed that this can be done when the algorithms will be implemented in the Test Programming Language. The picture acquisition problem has not been solved yet due to a lack of time, but recommendations concerning this subject are given. Related to the vision problem is the definition of the test patterns. This has been done roughly when the algorithms were defined. Details will be related to the solution for the vision problem.

Altogether it can be concluded that a AGAS has become available on which it is possible to realize an automated solution for the alignment of geometry of monitors and t.v. sets. Parts of the solution are already available and useful tips are given for the follow-up.
BIBLIOGRAPHY

[1] Holm, W.A. 
COLOUR TELEVISION EXPLAINED. 

[2] Schröter, F. 
VORSCHLÄGE FÜR EINE NEUE FERNSEHBILDSYNTHESE. 
Rheinisch-Westfälische Akademie der Wissenschaften.

MEETSYSTEEM MET 2-DIMENSIONALE CCD-ARRAY. 
CFT - report 59/77.

[4] Beisse, D. 
GEOMETRIEABGLEICH MIT KAMERA UND COMPUTER. 

[5] Spaan, A. 
EVALUATIE-RAPPORT CCD VIDEOCAMERA'S. 


MOIRE FRINGES IN STRAIN ANALYSIS. 

[8] Guild, J. 
DIFFRACTION GRATINGS AS MEASURING SCALES. 

MOIRE ANALYSIS OF STRAIN. 


PAPS - AN INTRODUCTION. 
CFT - report 09/83E.

[12] Jochems W. 
IGC-00 IBM-PC TO GPBS CONVERTER. 
Document number: 8122 284 24970

[13] INTER EBM SYSTEMS INFORMATION, volume I to VII. 
Rosier, W.J.
PICTURE ACQUISITION AND PROCESSING SYSTEM SOFTWARE.
Document numbers: 8122 968 1045.1
8122 968 1047.1
8122 968 1048.1
8122 968 1049.1
8122 968 1053.1

Rooy, J. de
VTG-20 - VIDEO TIMING GENERATOR SPECIFICATION.
Document number: 8122 284 2087.

Ponte, E. and S. van Raalte
COMPUTER CONTROLLED TV: THE DEFLECTION PROCESSOR 8432.
Laboratory report no: ETV 8612.

Hardware and Software Aspects of Computer Controlled TV.
IEEE Transactions on Consumer Electronics, Vol. CE-32, No. 1,
February 1986. by H. Wijnen.

Arnold, E.C.P.
IIC BUS CONTROL PROGRAM FOR IBM PC AND COMPATIBLE COMPUTERS.
Laboratory report no: TVE 87122.

Vonsee, R.G.
USER MANUAL OF MSPASCAL IIC BUS DRIVER.
Laboratory report no: TVE 87150.
APPENDIX A: POWER SUPPLY REQUIREMENTS

This appendix contains the data about the power every card needs and the modules that are used to create all power supplies.

Required power:

<table>
<thead>
<tr>
<th></th>
<th>+12VL</th>
<th>+5VL</th>
<th>-12VL</th>
<th>PS1(+5V)</th>
<th>PS2(+15V)</th>
<th>PS3(-15V)</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTG</td>
<td></td>
<td>0.7A</td>
<td></td>
<td>1.5A</td>
<td>0.2A</td>
<td>0.2A</td>
</tr>
<tr>
<td>VNE</td>
<td></td>
<td></td>
<td></td>
<td>0.6A</td>
<td>0.6A</td>
<td>0.3A</td>
</tr>
<tr>
<td>VPE</td>
<td></td>
<td></td>
<td></td>
<td>0.6A</td>
<td>0.6A</td>
<td>0.3A</td>
</tr>
<tr>
<td>VSE</td>
<td></td>
<td></td>
<td></td>
<td>0.7A</td>
<td>0.2A</td>
<td>0.2A</td>
</tr>
<tr>
<td>VAU</td>
<td>0.2A</td>
<td>2.1A</td>
<td>0.3A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PTB</td>
<td></td>
<td>2.5A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VNU</td>
<td></td>
<td>4.5A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAMERA</td>
<td>0.2A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IIC int.</td>
<td></td>
<td></td>
<td></td>
<td>0.1A</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>0.4A</td>
<td>9.8A</td>
<td>0.3A</td>
<td>3.5A</td>
<td>1.6A</td>
<td>1.0A</td>
</tr>
</tbody>
</table>

Power supplies used:

1x PE1112/02 (5V/18A)  
2x PE1113/02 (15V/6.0A)  
1x PE1143/02 (5V/7A, 12V/1A, 12V/1A)

To create:

- +5VL
- PS2, PS3
- PS1, +12VL, -12VL

Eventually for the EDS:

1x PE1114/02 (24V/2.9A)
APPENDIX B: FUNCTIONS INTERFACED TO \mu AMTS

GENERAL FUNCTIONS

genopc() : general opc routine
papsinit() : initialize PAPS

VAU FUNCTIONS

aain() : initialize a VAU
aaout() : allocate input data channel
abigr() : switch between binary and grey level
acuro() : switch cursor on/off
acurs() : cursor shift
agall() : get all parameters
agcur() : get cursor parameters
agina() : get input allocation
agota() : get output allocation
agrim() : get register image
agsam() : get sample pitch
agthr() : get threshold setting
agwin() : get window parameters
ainit() : initialize a VAU
asall() : set all parameters
ascur() : set cursor
asrim() : set register image
assam() : set sample pitch
asthr() : set threshold
aswin() : set window centre
asworg() : set window origin
athrs() : threshold shift
awinc() : window change
awino() : switch window on/off
awins() : window shift
awpie() : wait for picture enable

PTB FUNCTIONS

tinit() : initialize a PTB
tain() : allocate input data channel
tgina() : get input allocation
tsdrr() : set data rate restriction
tgdr() : get data rate restriction
tload() : load PTB with data, write pointer is reset
tloadw() : load PTB with data, write pointer is set
tivpnt() : input write pointer
tgmode() : get load mode
twait() : wait until ready
trdsli() : read data, one dimensional, integers
trds2b() : read data, two dimensional, bytes
RTR FUNCTIONS

rinit() : initialize an RTR
rain() : allocate input data channel
rgina() : get input allocation
raout() : allocate card outputs
rgota() : get output allocation
rgall() : get all parameter settings
rgrim() : get register image
rgsam() : get sample factor
rgthr() : get threshold
rrload() : load set in RTR
rsall() : set all parameters
rssam() : set sample factor
rsth() : set threshold

HCR FUNCTIONS

pinst() : input of status word
poutal() : set output allocation
prer() : read external register
prerr() : read random registers
prers() : read sequential registers
prersa() : read sequential registers, auto incrementing
pwer() : write external register
pwerb() : write external register bits
pwer() : write random registers
pwers() : write sequential registers
pwersa() : write sequential registers, auto increment
pwersi() : write sequential registers, indexed

VTG FUNCTIONS

Vtg_Initialise() : initialise VTG addressing in software
Vtg_Reset() : set default settings on VTG
Vtg_Store() : store current settings
Vtg_Restore() : set VTG according to stored settings
Vtg_Setup() : set VTG parameters
Vtg_Getpar() : get VTG settings

IGCR FUNCTIONS

INITGPBS() : initialize IGC-00 address in software
CLEARN() : reset GPBS bus
WER() : write external register
RER() : read external register
OTR_CIO() : output / control I/O
INR_TST_SST() : input / status control

IIC FUNCTIONS

IIC_TRANSCEIVE() : send or receive IIC data
IIC_TRACE() : trace IIC bus for address and data
**ADD FUNCTIONS**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOSDIR()</td>
<td>execute 'dir' command</td>
</tr>
<tr>
<td>DSHELL()</td>
<td>enter DOS temporarily</td>
</tr>
<tr>
<td>PTIME()</td>
<td>print time on the screen</td>
</tr>
<tr>
<td>Peek ()</td>
<td>input integer from memory</td>
</tr>
<tr>
<td>Poke ()</td>
<td>output integer to memory</td>
</tr>
<tr>
<td>Bpeek ()</td>
<td>input byte from memory</td>
</tr>
<tr>
<td>Bpoke ()</td>
<td>output byte to memory</td>
</tr>
<tr>
<td>Input ()</td>
<td>input integer from port</td>
</tr>
<tr>
<td>Output ()</td>
<td>output integer to port</td>
</tr>
<tr>
<td>Inchar ()</td>
<td>input byte from port</td>
</tr>
<tr>
<td>Outchar ()</td>
<td>output byte to port</td>
</tr>
</tbody>
</table>
APPENDIX C: DEFAULT SETTINGS

This appendix describes the default settings of the cards that are relevant to the control algorithms.

VAU
- Window is switched on;
- Cursor is switched off;
- Set to put grey values on the PI bus;
- Horizontal sample pitch = 1;
- Vertical sample pitch = 1;
- Input channel is connected to channel 1 on the PI-bus;
- Output channel is connected to channel 1 on the PI-bus;

PTB
- Data rate restriction = 0;
- PTB write pointer is reset;
- Load mode = 1;
- Input channel is connected to channel 1 on the PI-bus;