

**BRITISH STANDARD****BS EN
28632-1 : 1992
ISO 8632-1 :
1987****Specification for****Computer graphics: metafile for
the storage and transfer of picture
description information (CGM)****Part 1. Functional specification**

The European Standard EN 28632-1 : 1992 has the status of a
British Standard

Infographie: métafichier de stockage et de
transfert des informations de description
d'images — Spécifications
Partie 1. Spécification fonctionnelle

Graphische Systeme der Datenverarbeitung;
Datei für Speicherung und Übertragung von
Bildinformation
Teil 1. Funktionsbeschreibung

UDC 681.3.04/05 : 681.3.072.2(074)

NO COPYING IN ANY FORM WITHOUT WRITTEN PERMISSION FROM BSI

BSI
Standards
Documents

Document Sharing Hub

BS 6945 : Part 1 : 1988

National foreword

This Part of BS 6945 has been prepared under the direction of the Information Systems Technology Standards Committee. It is identical with ISO 8632-1 : 1987 'Information processing systems — Computer graphics — Metafile for the storage and transfer of picture description information Part 1: Functional specification' including Amendment 1

 published by the International Organization for Standardization (ISO). This Part of BS 6945 supersedes DD 107 : Part 1 : 1986, which is withdrawn.

 In 1992 the European Committee for Standardization (CEN) accepted ISO 8632-1 : 1987 and Addendum 1 : 1990 as European Standard EN 28632-1 : 1992. As a consequence of implementing the European Standard this British Standard is renumbered as BS EN 28632-1 and any reference to BS 6945 : Part 1 should be read as a reference to BS EN 28632-1.

The UK actively participated in the development of ISO 8632-1 through BSI Technical Committee IST/21, and the activity of its Computer Graphics Panel IST/21/2. The full list of organizations who have taken part in the work of the Technical Committee is given on the back cover. The Chairman of the Committee is Mr B. M. Wood, and Mr C. D. Osland has made a particular contribution to the drafting of the standard. Many of the UK experts serving on IST/21/2 and making important contributions to the international standard are not representatives of the cooperating organizations listed. BSI Committee IST/21 gratefully acknowledges both the personal contributions of the members of the Computer Graphics Panel and the cooperation of their employers.

Terminology and conventions. The text of the international standard has been approved as suitable for publication as a British Standard without deviation. Some terminology and certain conventions are not identical with those used in British Standards; attention is drawn especially to the following.

Wherever the words 'International Standard' appear, referring to this standard, they should be read as 'Part of BS 6945'. Wherever the words 'ISO 8632' appear, they should be read as 'BS 6945'.

Cross-references

International standard	Corresponding British Standard
ISO 646 : 1983	BS 4730 : 1985 Specification for UK 7-bit coded character set (Identical exercising national options)
ISO 7942 : 1985	BS 6390 : 1985 Specification for a set of functions for computer graphics programming, the Graphical Kernel System (GKS) (Identical)
ISO 2022 : 1986	BS 6856 : 1987 Specification for code extension techniques for United Kingdom 7-bit and 8-bit coded character sets (Identical)

The Technical Committee has reviewed the provisions of ISO 2375, to which reference is made in the text, and has decided that they are acceptable for use in conjunction with this standard.

Up-to-date information on existing and projected BSI publications for computer graphics is included in the 'Basic List', published as part of the BSI Information Technology Services (BITS) initiative. For details contact: Marketing Department, BSI, Linford Wood, Milton Keynes, MK14 6LE.

Compliance with a British Standard does not of itself confer immunity from legal obligations.



EUROPEAN STANDARD
NORME EUROPÉENNE
EUROPÄISCHE NORM

EN 28632-1

April 1992

UDC 681.3.04/.05 : 681.3.072.2(074)

Descriptors: Data processing, graphic data processing, information interchange, data handling, data transfer, images, description, files, specifications

English version

Information processing systems — Computer graphics —
Metafile for the storage and transfer of
picture description information
Part 1: Functional specification

(ISO 8632-1 : 1st edition, 1987 and its Addendum 1, 1990)

Systèmes de traitement de l'information —
Infographie — Métafichier de stockage et de
transfert des informations de description d'images
Partie 1: Description fonctionnelle
(ISO 8632-1 : 1ère édition, 1987 et son
Additif 1, 1990)

Informationsverarbeitungssysteme —
Computergraphik — Metadatei für die
Speicherung und die Übertragung von
Bildbeschreibungsinformationen
Teil 1: Funktionsbeschreibung
(ISO 8632-1 : 1. Ausgabe, 1987 und
Addendum 1, 1990)

This European Standard was approved by CEN on 1992-04-11. CEN members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the Central Secretariat or to any CEN member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CEN member into its own language and notified to the Central Secretariat has the same status as the official versions.

CEN members are the national standards bodies of Austria, Belgium, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland and United Kingdom.

CEN

European Committee for Standardization
Comité Européen de Normalisation
Europäisches Komitee für Normung

Central Secretariat: rue de Stassart 36, B-1050 Brussels

© CEN 1992 Copyright reserved to CEN members

Ref. No. EN 28632-1 : 1992 E



Documents

Document Sharing Hub

**Page 2****EN 28632-1 : 1992****Foreword**

The Technical Board has decided to submit the International Standard 8632-1 : 1987 and its addendum 1, 1990 *Information processing systems — Computer graphics — Metafile for the storage and transfer of picture description information — Part 1 : Functional specification* (ISO 8632-1 : 1987 and its addendum 1, 1990) for Formal Vote. The standard was accepted.

However the German version of the international standard is not available at the time of publication of the present European Standard.

National standards identical to this European Standard shall be published at the latest by 92-10-31 and conflicting national standards shall be withdrawn at the latest by 92-10-31.

According to the CEN/CENELEC Common Rules the following countries are bound to implement this standard: Austria, Belgium, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland and United Kingdom.



INTERNATIONAL STANDARD

ISO 8632-1

First edition
1987-08-01

AMENDMENT 1
1990-11-01

Information processing systems — Computer graphics — Metafile for the storage and transfer of picture description information —

Part 1 : Functional specification

AMENDMENT 1

*Systèmes de traitement de l'information — Infographie — Métafichier de stockage
et de transfert des informations de description d'images —*

Partie 1: Description fonctionnelle

AMENDEMENT 1



Reference number
ISO 8632-1 : 1987/Amd. 1 : 1990 (E)

AVA Documents
Document Sharing Hub



Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

- 4 In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO 8632-1/Amd. 1 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

© ISO 1990

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

International Organization for Standardization

Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

Contents

	Page		Page
National foreword		Figures	
Committees responsible	Inside front cover Back cover	1 VDC EXTENT establishes the direction of positive and negative angles	14
Specification		2 Anisotropic scaling of an ellipse	21
0 Introduction	2	3 Font description coordinate system	24
1 Scope and field of application	5	4 CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR	28
2 References	6	5 CHARACTER SPACING	29
3 Definitions and abbreviations	7	6 CHARACTER ORIENTATION	30
4 Concepts	10	7 CHARACTER HEIGHT and CHARACTER ORIENTATION after anisotropic transformation	31
5 Abstract specification of elements	43	8 Discrete text alignment with appended text and proportional spacing	34
6 Metafile defaults	102	9 Discrete text alignment	35
7 Conformance	105	10 Continuous text alignment	36
Annexes		11 Continuous text alignment after anisotropic transformation	37
A Formal grammar of the functional specification	106	12 State diagram	42
B Guidelines for private encodings	119	13 Example for POLYGON SET	69
C Reference models	120	14 nx-by-nx rectangle mapped onto display surface	71
D Guidelines for metafile generators and interpreters	124	15 CIRCULAR ARC 3 POINT CLOSE specifica- tions with 'pie' and 'chord'	75
E Relationship of CGM and GKS	135	16 Relationship of CGM to tradition 1 graphics package	121
Tables		17 Metafile interpretation with no graphics package	122
1 Individual attributes and attributes that may be bundled	22	18 Metafile interpretation using traditional graphics package	123
2 Aspects of the bundle and affected primitives	23	19 Mapping of direct colour specification from metafile to device	127
3 Cases allowing display of partial text	26		
4 Data type definitions and abbreviations	43		
5 Suggested minimum capabilities	134		
6 Mapping of control functions	139		
7 Mapping of output functions	140		
8 Mapping of attribute functions	141		
9 Mapping of transformation functions	142		
10 Metafile descriptor contents in GKS environment	143		

British Standard Specification for

Computer graphics: metafile for the storage and transfer of picture description information (CGM)

Part 1. Functional specification

0 Introduction

0.1 Purpose

The Computer Graphics Metafile provides a file format suitable for the storage and retrieval of picture information. The file format consists of a set of elements that can be used to describe pictures in a way that is compatible between systems of different architectures and devices of differing capabilities and design.



This picture description includes the capability for describing static pictures. Static pictures are those where elements which may lead to dynamic effects (for example those leading to regeneration) are prohibited within the picture body.

0.2 Reasons for this International Standard

The main reasons for producing a standard computer graphics metafile are

- a) to allow picture information to be stored in an organized way on a graphical software system;
- b) to facilitate transfer of picture information between different graphical software systems;
- c) to enable picture information to be transferred between graphical devices;
- d) to enable picture information to be transferred between different computer graphics installations.

0.3 Design requirements

To reach these objectives, a number of design principles were adopted:

- a) The metafile should provide a suitable set of elements for the transfer of a wide range of pictorial information.
- b) The metafile should address the more usual and essential features found on graphical devices directly and should provide access to less common facilities via an escape mechanism.
- c) The design of the metafile should not preclude extension of ISO 8632 at a later stage to cover facilities beyond those included in this version of the Standard.

It should also not preclude further extensions to support future standards.

- d) The metafile should be usable from GKS (Graphical Kernel System - ISO 7942) with both metafile input and metafile output functions.

It should include the capability to support ISO 7942 (GKS) static picture-capture.

- e) ISO 8632 should address the needs of different applications that have conflicting requirements for size of metafile, speed of generation and interpretation, readability, editability and ease of transfer through different transport mechanisms.

0.4 Design criteria

The requirements of 0.3 were used to formulate the following criteria which were used to decide between different design possibilities.

- a) Completeness: In any area of ISO 8632, the functionality specified by ISO 8632 should be complete in itself.
- b) Conciseness: Redundant elements or parameters should be avoided.
- c) Consistency: Contradictory elements should be avoided.
- d) Extensibility: The ability to add new elements and generality to ISO 8632 should not be precluded.
- e) Fidelity: The minimal results and characteristics of elements should be well defined.



Documents
Document Sharing Hub

Design criteria**Introduction**

- f) **Implementability:** An element should be able to be supported efficiently on most host systems and/or graphics hardware.
- g) **Orthogonality:** The elements of the metafile should be independent of each other, or any dependencies should be structured and well defined.
- h) **Predictability:** ISO 8632 should be such that the recommended or proper use of standard elements guarantees the results of using a particular element.
- i) **Standard practice:** Only those elements that reflect existing practice, that are necessary to support existing practice, or that are necessary to support proposed standards should be standardized.
- j) **Usefulness:** Functions should be powerful enough to perform useful tasks.
- k) **Well-structured:** The assumptions that elements make about each other should be minimized. An element should have a well-defined interface and a simply stated unconditional purpose. Multipurpose elements and side effects should be avoided.

0.5 Access to a metafile

The metafile has been designed so that, although its main usage is anticipated as being with completely sequential access, non-sequential access is also possible. Once the basic environment of the metafile has been established, individual pictures may be accessible if the medium, the encoding and the implementation support this form of access.

0.6 Generation and interpretation of metafiles

The specific mechanisms of metafile generation and interpretation are not described by ISO 8632, although it does describe the intended result of such interpretation. The basic set of metafile elements includes a capability for the addition of application-dependent data, which do not have graphical meaning and for which no intended interpretation results are described.

0.7 Distinction between formal specification and encodings

The functionality provided by the metafile is separated from the specification of any particular encoding format. ISO 8632 provides for both standard and private encodings of the elements described in this part of ISO 8632. Guidelines for private encodings are specified in annex B; these guidelines are do not form part of ISO 8632.

Three standard encodings are specified in parts 2, 3 and 4 ISO 8632. Each of the standardized encodings is capable of representing the full functionality described in this part of ISO 8632. Translation between the standardized encodings is possible without loss of picture information, although subsequent translation back into the original encoding may not result in precisely the same data stream, due to different quantizations of precisions in the different encodings.

The character encoding specified in ISO 8632/2 is intended to provide an encoding of minimum size. It conforms to the rules for code extension specified in ISO 2022 in the category of complete code system. It is particularly suitable for transfer through networks that cannot support binary transfers.

The binary encoding specified in ISO 8632/3 provides an encoding that requires least effort to generate and interpret on many systems.

The clear text encoding specified in ISO 8632/4 provides an encoding that can be created, viewed and edited with standard text editors. It is therefore also suitable for transfer through networks that support only transfer of text files.

Introduction**Relationship to other International Standards****0.8 Relationship to other International Standards**

ISO 8632 draws extensively for its model of a graphics system on GKS (Graphical Kernel System — ISO 7942). In addition, ISO 8632 specifies a metafile that may be used as a static picture-capture metafile by GKS. One relationship between ISO 8632 and GKS — the use of a subset of the elements of this Standard as a static picture capture metafile by GKS — is explained in annex E.

The character encoding specified in ISO 8632/2 conforms to the code extension techniques of ISO 2022.

The binary encoding specified in ISO 8632/3 employs the mechanism for representing floating point numbers specified in ANSI/IEEE 754-1986.

For certain elements, the CGM defines value ranges of parameters as being reserved for registration. The meanings of these values will be defined using the established procedures (see 4.11) of the ISO International Registration Authority for Graphical Items. These procedures do not apply to values and value ranges defined as being reserved for private use; these values and ranges are not standardized.



There is a very close relationship between many of the elements in ISO 8632 and a subset of the functions in the CGI (Computer Graphics Interface - ISO/IEC 9636 (currently a Draft International Standard)).

0.9 Status of annexes

The annexes do not form an integral part of this part of ISO 8632 but are included for information only.

1 Scope and field of application

ISO 8632 provides a file format suitable for the storage and retrieval of picture description information. The file format consists of an ordered set of elements that can be used to describe pictures in a way that is compatible between systems of different architectures and devices of differing capabilities and design.

This picture description includes the capability for describing static images.

The elements specified provide for the representation of a wide range of pictures on a wide range of graphical devices. The elements are split into groups that delimit major structures (metafiles and pictures), that specify the representations used within the metafile, that control the display of the picture, that perform basic drawing actions, that control the attributes of the basic drawing actions and that provide access to non-standard device capabilities.

The Metafile is defined in such a way that, in addition to sequential access to the whole metafile, random access to individual pictures is well-defined; whether this is available in any system that uses ISO 8632 depends on the medium, the encoding and the implementation.

In addition to a functional specification, three standard encodings of the metafile syntax are specified. These encodings address the needs of applications that require minimum metafile size, minimum effort to generate and interpret, and maximum flexibility for a human reader or editor of the metafile.

This part of ISO 8632 describes the format using an abstract syntax. The remaining three parts of ISO 8632 specify three standardized encodings that conform to this syntax: part 2 specifies a character encoding that conforms to the rules for code extension specified in ISO 2022 in the category of complete coding system; part 3 specifies a binary encoding; part 4 specifies a clear text encoding.



2 References

ISO 646, *Information processing—ISO 7-bit coded character set for information interchange.*

ISO 2022, *Information processing—ISO 7-bit and 8-bit, coded character sets—Code extension techniques.*

ISO 2375, *Data processing—Procedure for registration of escape sequences.*

ISO 7942, *Information processing systems—Computer graphics—Graphical Kernel System (GKS) functional description.*



ISO/IEC 9636 *Information processing systems - Computer Graphics - Interfacing techniques for dialogues with graphical devices (CGI). Parts 1-6 (currently a Draft International Standard).*

3 Definitions and abbreviations

3.1 Definitions

For the purposes of this International Standard the following definitions apply.

NOTE — As far as possible, commonly accepted graphics terminology is used.

3.1.1 aspect ratio: The ratio of the width to the height of a rectangular area, such as a window or viewport. For example, an aspect ratio of 2.0 indicates an area twice as wide as it is high.

3.1.2 aspect source flag (ASF): Indicator as to whether a particular attribute selection is to be individual or bundled.

3.1.3 aspects of primitives: Ways in which the appearance of a primitive can vary. Some aspects are controlled directly by primitive attributes; some are controlled indirectly through a bundle table.

3.1.4 attribute elements: Metafile elements that describe the appearance of graphical elements.

3.1.5 bundle: Set of attributes associated with one of the following graphical element types: line, marker, text, and filled area.

3.1.6 bundle index: An index into a bundle table for a particular output primitive.

3.1.7 bundle table: An indexed table containing a set of attributes for each index.

3.1.8 clip indicator: Indicator as to whether metafile graphical elements are to be clipped at the limits of CLIP RECTANGLE.

3.1.9 clip rectangle: A rectangle defined in VDC space which is used as a clipping boundary when the metafile graphical elements are to be clipped.

3.1.10 clipping: Removing parts of display elements that lie outside a given boundary.

3.1.11 colour selection mode: Indicator as to whether colour selection is to be direct (by specifying the RGB values) or indexed (by specifying an index into a table of RGB values).

3.1.12 colour table: A table for use in mapping from a colour index to the corresponding colour. See DIRECT COLOUR, INDEXED COLOUR.

3.1.13 colour value: The values of the RGB (red, green, blue) components describing a colour.

3.1.14 Computer Graphics Interface (CGI): The specification for interface techniques for dialogues with graphical devices.

3.1.15 Computer Graphics Metafile (CGM): The specification for a mechanism for storing and transferring picture description information.

3.1.16 conjugate diameter pair (CDP): A pair D, d of diameters of an ellipse such that a tangent to the ellipse at each endpoint of one diameter is parallel to the other diameter.

3.1.17 control elements: Metafile elements that specify address space, clipping boundaries, picture delimiters, and format descriptions of the Metafile elements.

3.1.18 data interface: An interface between software modules or devices comprising one or more packets containing opcodes and data—as contrasted with a subroutine call interface.

3.1.19 descriptor elements: Metafile elements that describe the functional content, format, default conditions, identification, and characteristics of the Metafile.

3.1.20 device driver: The device-dependent part of a graphics implementation which supports a physical device. The device driver generates device-dependent output.

3.1.21 direct colour: A colour selection scheme in which the colour values are specified directly, without requiring an intermediate mapping via a colour table. See COLOUR TABLE, INDEXED

Definitions and abbreviations

Definitions

COLOUR.

3.1.22 display surface: That part of a graphics device upon which a visible image appears (for example, the screen of a display, the paper in a plotter).

3.1.23 escape elements: Metafile elements that describe device- or system-dependent elements used to construct a picture, but that are not otherwise standardized.

3.1.24 external elements: Metafile elements that communicate information not directly related to the generation of a graphical image.

3.1.25 font: The typeface or style of characters, independent of other text attributes such as size and rotation. The font is distinct from the character set.

3.1.26 graphical primitive elements: Metafile elements that describe images in the Metafile.



3.1.27 Graphical Kernel System (GKS): A standardized application programmer's interface to graphics.

3.1.28 graphics device: A device (for example, refresh display, storage tube display, or plotter) on which display images can be represented.

3.1.29 hatch style: A format for filling closed figures. A hatch style consists of one or more sets of lines whose presence represents the interior of the figure in question.

3.1.30 indexed colour: A colour selection scheme in which the colour index is used to retrieve colour values from a colour table. See COLOUR TABLE, DIRECT COLOUR.

3.1.31 message: A string of characters used to communicate information to operators at Metafile interpretation time.

3.1.32 metafile: A mechanism for retaining and transporting graphical data and control information. This information contains a device-independent description of one or more pictures.

3.1.33 Metafile Descriptor (MD): A metafile element that describes the format of the metafile (but not its encoding method) and the functionality expected of a metafile interpreter.

3.1.34 metafile element: A functional item that can be used to construct a picture or convey information.

3.1.35 metafile generator: The process or equipment that produces the Computer Graphics Metafile.

3.1.36 metafile interpreter: The process or equipment that reads the Computer Graphics Metafile and interprets the contents. An interpreter may be needed in order to drive a Computer Graphics Interface or other device interface to obtain a picture that resembles the intended picture as closely as possible.

3.1.37 normalized device coordinates (NDC): Coordinates specified in a device-independent coordinate system, normalized to some range (typically 0 to 1). See VDC EXTENT, VDC RANGE, VDC SPACE, VIRTUAL DEVICE COORDINATES.

3.1.38 pattern style: A format for filling closed figures with patterns. A pattern style consists of an array of variously coloured or shaded cells.

3.1.39 Picture Descriptor (PD): A set of metafile elements used to set the interpretation modes of attribute elements for the entire picture.

3.1.40 pixel: The smallest element of a display surface that can be independently assigned colour.

3.1.41 realized edge: The zero-width ideal boundary line of the filled-area if the edge is invisible, and the finite-width displayed line if the edge is visible.

3.1.42 realized interior: In a filled area element, that portion of the ideal interior as extending to and terminating at the realized edge.

3.1.43 view surface: See DISPLAY SURFACE.

3.1.44 virtual device: An idealized graphics device that presents a set of graphics capabilities to graphics software or systems via the Computer Graphics Interface.

Definitions

Definitions and abbreviations

- 3.1.45 virtual device coordinates (VDC):** The coordinates used to specify position in the VDC space. These are absolute two-dimensional coordinates. See VDC SPACE.
- 3.1.46 VDC extent:** A rectangular region of interest contained within the VDC range. See VDC RANGE, VDC SPACE.
- 3.1.47 VDC range:** A rectangular region within VDC space consisting of the set of all coordinates representable in the declared coordinate type, precision, and encoding format of the metafile. See VDC EXTENT, VDC SPACE.
- 3.1.48 VDC space:** A two-dimensional Cartesian coordinate space of infinite precision and extent. Only a subset of VDC space, the VDC range, is realizable in a metafile. See VDC EXTENT, VDC RANGE, VIRTUAL DEVICE COORDINATES.
- 3.1.49 anisotropic mapping:** A mapping in which the scale factors applied along each axis are not equal. This is often used in reference to the mapping from VDC to distance units on the physical display surface. With anisotropic mapping, the angle between any pair of non-parallel line segments can change; circles cease to be circles and become post-transformed ellipses. See "isotropic mapping".
- 3.1.50 boundary:** The mathematical locus that defines, in abstract VDC space, the limits of a region to be filled (for fill primitives and closed figures). The visual appearance of interior style 'hollow' consists of a depiction of the boundary obtained after clipping has been taken into account.
- 3.1.51 character set:** The set of displayable symbols mapped to individual characters in a TEXT, APPEND TEXT, or RESTRICTED TEXT string. This corresponds to the "G-set" defined in ISO 2022. A character set is independent of the font or typeface; examples of character sets are: ASCII (X3.4), German and Katakana.
- 3.1.52 clipping mode:** A generic term referring to one of Line Clipping, Marker Clipping or Edge Clipping Modes. An object clipping may be either 'locus', 'shape' or 'locus then shape'.
- 3.1.53 closed figure:** A compound primitive that behaves as a fill primitive of more general shape. It is formed by bracketing a sequence of line or fill primitives, edge attributes, and certain control elements, with the elements BEGIN FIGURE and END FIGURE.
- 3.1.54 compound primitive:** A compound primitive is specified by a sequence of CGM elements, as opposed to primitives represented by a single element. Compound text and closed figures are examples of compound primitives in the CGM.
- 3.1.55 compound text:** A compound text primitive is formed through the use of APPEND TEXT. There may be attribute changes between portions of the resulting complete text string.
- 3.1.56 device coordinates:** The coordinates native to a device; device-dependent coordinates; physical device coordinates.
- 3.1.57 device viewport:** A rectangular subset of the physical display surface into which VDC EXTENT is mapped. See "effective viewport".
- 3.1.58 edge:** The rendering of the perimeter of a filled region, controlled by edge attributes. Edges are clipped after being applied to the boundary, as distinct from the rendition of the boundary obtained from interior style 'hollow'. See "boundary".
- 3.1.59 effective viewport:** The actual viewport resulting from forced isotropic mapping from the VDC extent to the viewport.
- 3.1.60 foreground colour:** The colour used in the rendering process in which primitives are rendered on the display surface, as opposed to the BACKGROUND COLOUR or AUXILIARY COLOUR. The foreground colour is set separately for each class of primitive.
- 3.1.61 global segment:** A segment that is defined in the Metafile Descriptor (see "segment"). It may be referenced from within any picture.
- 3.1.62 graphic object:** A graphic object is a graphic primitive, including a compound primitive, together with the associated attributes.
- 3.1.63 isotropic mapping:** A mapping which is invariant with respect to direction; equal scaling in all orthogonal representational dimensions. It is often used to describe the mapping from VDC to distance units on the physical display surface. With isotropic mapping, the angle between any pair of non-parallel line segments remains unchanged; for example, circles remain circles. See "anisotropic mapping".

1

3.1.64 local segment: A segment whose definition is local to the picture in which it appears.

3.1.65 object clipping: Object clipping is applied to a graphic object. For example, clipping is applied to a line after it has had the width attribute associated with it.

3.1.66 region: In the context of closed figures or the POLYGON SET element, an area that is explicitly or implicitly closed, that is a subset of the full area being filled. Regions can be nested, disjoint or overlapping. The boundaries of all regions are considered together when applying the interior test for filling a closed figure or POLYGON SET.

3.1.67 segment: A collection of primitives, primitive attributes and some additional attributes associated with the segment as a whole. See "segment attribute".

3.1.68 segment attribute: An attribute associated with a segment as a whole rather than attributes of individual primitives.

3.1.69 size specification mode: A generic term for Line Width Specification Mode, Edge Width Specification Mode, or Marker Size Specification Mode. A size specification mode may be 'absolute' or 'scaled', the latter being referenced to a nominal size in device coordinate space.

3.1.70 skewed: Used to describe stroke precision text when the CHARACTER ORIENTATION vectors are non-perpendicular; CELL ARRAYs when the three defining points form a parallelogram which is not a rectangle; or a segment transformation that causes rectangles to become non-rectangular parallelograms.

3.2 Abbreviations

The following abbreviations are used in all parts of ISO 8632.

ASF	Aspect Source Flag
CDP	Conjugate Diameter Pair
CGI	Computer Graphics Interface
CGM	Computer Graphics Metafile
GKS	Graphical Kernel System
MD	Metafile Descriptor
NDC	Normalized Device Coordinate(s)
PD	Picture Descriptor
VDC	Virtual Device Coordinate(s)

4.1 Introduction

The objective of the Computer Graphics Metafile (CGM) is to provide for the description, storage, and communication of graphical information in a device-independent manner. To accomplish this, ISO 8632 defines the form (syntax) and functional behaviour (semantics) of a set of elements that may occur in the CGM. The following classes of elements are defined:

- Delimiter Elements, which delimit significant structures within the Metafile.
- Metafile Descriptor Elements, which describe the functional content, default conditions, identification, and characteristics of the CGM.
- Picture Descriptor Elements, which set the interpretation modes of attribute elements for each picture.
- Control Elements, which allow picture boundaries and coordinate representation to be modified.
- Graphical Primitive Elements, which describe the visual components of a picture in the CGM.
- Attribute Elements, which describe the appearance of graphical primitive elements.
- Escape Element, which describes device- or system-dependent elements used to construct a picture; however, the elements are not otherwise standardized.
- External Elements, which communicate information not directly related to the generation of a graphical image.

1

Segment Elements, which enable the grouping and manipulation of elements.



Documents

Document Sharing Hub

A Computer Graphics Metafile is a collection of elements from this standardized set. The BEGIN METAFILE and END METAFILE elements each occur exactly once in a complete metafile; as many or as few of the elements in the other classes may occur as are needed. A metafile needs to be interpreted in order to display its pictorial content on a graphics device. The Descriptor Elements give the interpreter sufficient data to interpret metafile elements and to make informed decisions concerning the resources needed for display.

Any CGM contains certain delimiter elements; in addition it may include control elements for metafile interpretation, Picture Descriptor elements for declaring parameter modes of attribute elements, graphical primitive elements for defining graphical entities, attribute elements for defining the appearance of the graphical primitive elements, escape elements for accessing non-standardized features of particular devices, and external elements for communication of information external to the definition of the pictures in the CGM.



Graphical output primitives and attributes may be grouped in segments. Segment attribute elements control the appearance of segments.

A minimal correct metafile consists of BEGIN METAFILE, a Metafile Descriptor consisting of METAFILE VERSION and METAFILE ELEMENT LIST, and END METAFILE.

4.2 Delimiter elements

Every metafile starts with a BEGIN METAFILE element and ends with an END METAFILE element. This allows multiple metafiles to be stored or transferred together.

Each picture starts with a BEGIN PICTURE element and ends with an END PICTURE element. Between these delimiters, the Picture Descriptor is separated from the picture body by a BEGIN PICTURE BODY element.

Once the Metafile Descriptor has been read, access to individual pictures, on a random as opposed to sequential basis, may be safely accomplished if the encoding, access mechanism and implementation permit.

Delimiter elements

Concepts

BEGIN METAFILE and BEGIN PICTURE both have parameters for a name by which the metafile and picture (respectively) can be identified.



Primitives may be grouped together to form a composite primitive known as a closed figure. The primitives to be included in the closed figure being defined are delimited by the elements BEGIN FIGURE and END FIGURE.

Groups of elements, called segments, are delimited by BEGIN SEGMENT and END SEGMENT. Each segment is uniquely identified by a segment identifier. Segments may be defined in the Metafile Descriptor or within picture bodies.

Concepts

4.3 Metafile descriptor elements

The Metafile Descriptor (MD) is a group of elements that describes the functional capabilities required to interpret the CGM. These elements are

METAFILE VERSION	MAXIMUM COLOUR INDEX
METAFILE DESCRIPTION	COLOUR VALUE EXTENT
VDC TYPE	METAFILE ELEMENT LIST
INTEGER PRECISION	METAFILE DEFAULTS
REAL PRECISION	REPLACEMENT
INDEX PRECISION	FONT LIST
COLOUR PRECISION	CHARACTER SET LIST
COLOUR INDEX PRECISION	CHARACTER CODING ANNOUNCER
NAME PRECISION	
MAXIMUM VDC EXTENT	
SEGMENT PRIORITY EXTENT	

NOTE - Other elements, as defined in this part of ISO/IEC 8632, may appear within the Metafile Descriptor within the definition of a global segment.

In a particular metafile, the METAFILE ELEMENT LIST lists at least those standardized elements that occur in the metafile. The CGM interpreter is thus informed of the capabilities required to successfully interpret the Computer Graphics Metafile. The CGM contains a single Metafile Descriptor. The Metafile Descriptor immediately follows the BEGIN METAFILE element in a metafile (with the possible exception of intervening external and escape elements).

METAFILE VERSION and METAFILE ELEMENT LIST shall occur only once in the Metafile Descriptor for version 1 metafiles. It is recommended that they shall only appear once in version 1 metafiles.

NOTE - It is recommended that the following elements: METAFILE VERSION, METAFILE ELEMENT LIST and (possibly multiple occurrences of) METAFILE DESCRIPTION appear first in the Metafile Descriptor and in the order listed.

4.3.1 Identification

The identifying information includes declaration of the version of ISO 8632 and descriptive information about the origin, owner, generation date, etc., of the metafile.

4.3.2 Functional capability

The contents of the Computer Graphics Metafile are defined by the METAFILE ELEMENT LIST element. This contains a list of the control elements, graphical primitive elements, and attribute elements that are utilized in the metafile. Several shorthand names for CGM elements are also provided for use with the METAFILE ELEMENT LIST. The shorthand names shall not be considered macro names, nor shall they be construed to be levels of conformance.

4.3.2.1 Drawing set

The drawing set includes the mandatory CGM elements (i.e., those that shall appear in every conforming CGM) and most of the graphical primitive elements and attribute elements. The drawing set is specified by the shorthand name DRAWING SET.

The elements included in the drawing set are:

BEGIN METAFILE	LINE TYPE
END METAFILE	LINE WIDTH
BEGIN PICTURE	LINE COLOUR
BEGIN PICTURE BODY	MARKER BUNDLE INDEX
END PICTURE	MARKER TYPE
METAFILE VERSION	MARKER SIZE
METAFILE DESCRIPTION	MARKER COLOUR
VDC TYPE	TEXT BUNDLE INDEX
METAFILE ELEMENT LIST	TEXT FONT INDEX
AUXILIARY COLOUR	TEXT PRECISION

Concepts

TRANSPARENCY
 CLIP RECTANGLE
 CLIP INDICATOR
 VDC EXTENT
 BACKGROUND COLOUR
 COLOUR SELECTION MODE
 POLYLINE
 DISJOINT POLYLINE
 POLYMARKER
 TEXT
 RESTRICTED TEXT
 APPEND TEXT
 POLYGON
 POLYGON SET
 CELL ARRAY
 GENERALIZED DRAWING PRIMITIVE
 RECTANGLE
 CIRCLE
 CIRCULAR ARC 3 POINT
 CIRCULAR ARC 3 POINT CLOSE
 CIRCULAR ARC CENTRE
 CIRCULAR ARC CENTRE CLOSE
 ELLIPSE
 ELLIPTICAL ARC
 ELLIPTICAL ARC CLOSE
 LINE BUNDLE INDEX

Metafile descriptor elements

CHARACTER EXPANSION FACTOR
 CHARACTER SPACING
 TEXT COLOUR
 CHARACTER HEIGHT
 CHARACTER ORIENTATION
 TEXT PATH
 TEXT ALIGNMENT
 FILL BUNDLE INDEX
 INTERIOR STYLE
 FILL COLOUR
 HATCH INDEX
 PATTERN INDEX
 EDGE BUNDLE INDEX
 EDGE TYPE
 EDGE WIDTH
 EDGE COLOUR
 EDGE VISIBILITY
 FILL REFERENCE POINT
 PATTERN TABLE
 PATTERN SIZE
 COLOUR TABLE
 ASPECT SOURCE FLAGS
 ESCAPE
 MESSAGE
 APPLICATION DATA

4.3.2.2 Drawing plus control set

The drawing-plus-control set may be used to indicate all of the elements in the drawing set plus additional control, Metafile Descriptor, Picture Descriptor, and attribute elements. It is specified by the shorthand name DRAWING PLUS CONTROL SET.

The elements included in the drawing-plus-control set are all of the elements in the drawing set and the following elements:

INTEGER PRECISION
 REAL PRECISION
 INDEX PRECISION
 COLOUR PRECISION
 COLOUR INDEX PRECISION
 MAXIMUM COLOUR INDEX
 COLOUR VALUE EXTENT
 METAFILE DEFAULTS REPLACEMENT
 FONT LIST
 CHARACTER SET LIST

CHARACTER CODING ANNOUNCER
 VDC INTEGER PRECISION
 VDC REAL PRECISION
 SCALING MODE
 LINE WIDTH SPECIFICATION MODE
 MARKER SIZE SPECIFICATION MODE
 EDGE WIDTH SPECIFICATION MODE
 CHARACTER SET INDEX
 ALTERNATE CHARACTER SET INDEX

4.3.2.3 Version 2 set

The Version-2 set may be used to indicate all the elements in the drawing-plus-control set and all the additional elements defined in this part of ISO/IEC 8632.

4.3.2.4 Extended primitives set

The extended-primitives set may be used to indicate those primitives which are not defined in ISO 7942 (GKS). These elements are:



DISJOINT POLYLINE
 RESTRICTED TEXT
 APPEND TEXT
 POLYGON SET
 RECTANGLE
 CIRCLE
 CIRCULAR ARC 3 POINT
 CIRCULAR ARC 3 POINT CLOSE
 CIRCULAR ARC CENTRE
 CIRCULAR ARC CENTRE CLOSE
 CIRCULAR ARC CENTRE REVERSED
 ELLIPSE
 ELLIPTICAL ARC
 ELLIPTICAL ARC CLOSE
 CONNECTING EDGE

4.3.2.5 Version 2 GKSM set

The Version-2-GKSM set includes elements for ISO 7942 (GKS) picture capture. The elements included in the Version-2-GKSM set are:

BEGIN METAFILE	NAME PRECISION
BEGIN PICTURE	MAXIMUM COLOUR INDEX
BEGIN PICTURE BODY	COLOUR VALUE EXTENT
END PICTURE	METAFILE ELEMENT LIST
BEGIN SEGMENT	METAFILE DEFAULTS REPLACEMENT
END SEGMENT	FONT LIST
END METAFILE	CHARACTER SET LIST
METAFILE VERSION	CHARACTER CODING ANNOUNCER
METAFILE DESCRIPTION	MAXIMUM VDC EXTENT
VDC TYPE	SEGMENT PRIORITY EXTENT
INTEGER PRECISION	VDC EXTENT
REAL PRECISION	DEVICE VIEWPORT
INDEX PRECISION	DEVICE VIEWPORT MAPPING
COLOUR PRECISION	DEVICE VIEWPORT SPECIFICATION MODE
COLOUR INDEX PRECISION	LINE REPRESENTATION
MARKER REPRESENTATION	CHARACTER HEIGHT
TEXT REPRESENTATION	CHARACTER ORIENTATION
FILL REPRESENTATION	TEXT PATH
VDC INTEGER PRECISION	TEXT ALIGNMENT
VDC REAL PRECISION	CHARACTER SET INDEX
CLIP RECTANGLE	ALTERNATE CHARACTER SET INDEX
POLYLINE	FILL BUNDLE INDEX
POLYMARKER	INTERIOR STYLE
TEXT	FILL COLOUR
POLYGON	HATCH INDEX
CELL ARRAY	PATTERN INDEX
GDP	FILL REFERENCE POINT
LINE BUNDLE INDEX	PATTERN TABLE
LINE TYPE	PATTERN SIZE
LINE WIDTH	COLOUR TABLE
LINE COLOUR	ASPECT SOURCE FLAGS
MARKER BUNDLE INDEX	PICK IDENTIFIER
MARKER TYPE	ESCAPE
MARKER SIZE	MESSAGE
MARKER COLOUR	APPLICATION DATA
TEXT BUNDLE INDEX	SEGMENT TRANSFORMATION
TEXT FONT INDEX	SEGMENT HIGHLIGHTING
TEXT PRECISION	SEGMENT DISPLAY PRIORITY
CHARACTER EXPANSION FACTOR	SEGMENT PICK PRIORITY
CHARACTER SPACING	
TEXT COLOUR	

4.3.3 Default metafile state

The default state is the state to which the interpreter is returned at the start of each picture. The default states of all metafile elements are defined in clause 6. These default values may be selectively replaced by using the METAFILE DEFAULTS REPLACEMENT element. The correspondence between character set indexes and registered or private character sets, and the meaning assigned to text font indexes, are also established in the Metafile Descriptor.

4.4 Picture descriptor elements

Picture Descriptor elements include elements to declare the parameter modes of other elements, to declare the entire picture, to configure that portion of coordinate space that is of interest in the picture, and to set the colour to which the view surface is cleared at the start of the picture. These elements are

Picture descriptor elements

Concepts

SCALING MODE, COLOUR SELECTION MODE, LINE WIDTH SPECIFICATION MODE, MARKER SIZE SPECIFICATION MODE, EDGE WIDTH SPECIFICATION MODE, VDC EXTENT, and BACKGROUND COLOUR. If included in a picture, they shall appear after the BEGIN PICTURE element and before the BEGIN PICTURE BODY element. Escape and external elements are permitted in the Picture Descriptor.

- 1 Some of the picture descriptor elements may appear outside the Picture Descriptor if this is permitted by the formal grammar for the metafile version. In such a case they do not set parameter values to apply for the entire picture.

4.4.1 Scaling mode

VDC space may be either an abstract space, which may be mapped to an arbitrary size on a physical device, or a metric space, which is intended to be mapped to a particular size. Selection of the mode to be used can be made on a picture-by-picture basis by means of the SCALING MODE element. The scaling mode element provides a flag to select abstract space or metric space, and a scale factor which specifies the number of millimeters per VDC unit when metric space is selected.

4.4.2 Colour selection mode

- 1 COLOUR SELECTION MODE selects either indexed or direct (RGB) colour specification and is described further under colour attributes. For version 1 metafiles the selection is for the whole picture.

4.4.3 Specification modes

Line width, marker size, and edge width may be specified in more than one way. The width of lines, for example, may be specified as either a measure in VDC units or as a scale factor to be applied to a device-dependent nominal line width at interpretation time. For each attribute element having such multiple modes, there is an associated control element that defines the mode of the parameter of the attribute element.

4.4.4 VDC extent

There is a metafile element to define the VDC extent. The extent is set with the VDC EXTENT element by specifying the addresses (in VDC) of the lower-left corner and the upper-right corner of this extent as seen by the viewer of the picture. Specification of values outside the VDC extent is permitted in CGM elements. It is intended that the visible portion of an image be contained within the VDC extent. It thus provides a frame for the region of interest in a picture. The values of the coordinates for either dimension may be either increasing or decreasing from the lower-left to the upper-right corner. For example, for devices with an upper-left origin, a picture may be described in coordinates that map directly to the device but still may be displayed correctly on a device with a lower-left origin. Figure 1 illustrates these concepts.

The VDC extent thus establishes the sense and orientation of VDC space (that is, the directions of the positive x (+x) and positive y (+y) axes, and whether the +y axis is 90° clockwise or 90° counterclockwise from the +x axis). In particular, VDC EXTENT establishes the direction of positive and negative angles as follows: positive 90° is defined to be the right angle from the positive x-axis to the positive y-axis (see figure 1). Note that some attributes such as text attributes (for example, the directions of the 'up' and 'base' component vectors of CHARACTER ORIENTATION, and therefore the meaning of the enumerative values 'right', 'left', 'up', 'down') are intimately bound to these definitions.

The default state of the extent is specified in clause 6 and can be changed in the METAFILE DEFAULTS REPLACEMENT element in the MD. VDC EXTENT returns to this default state at the beginning of each picture.

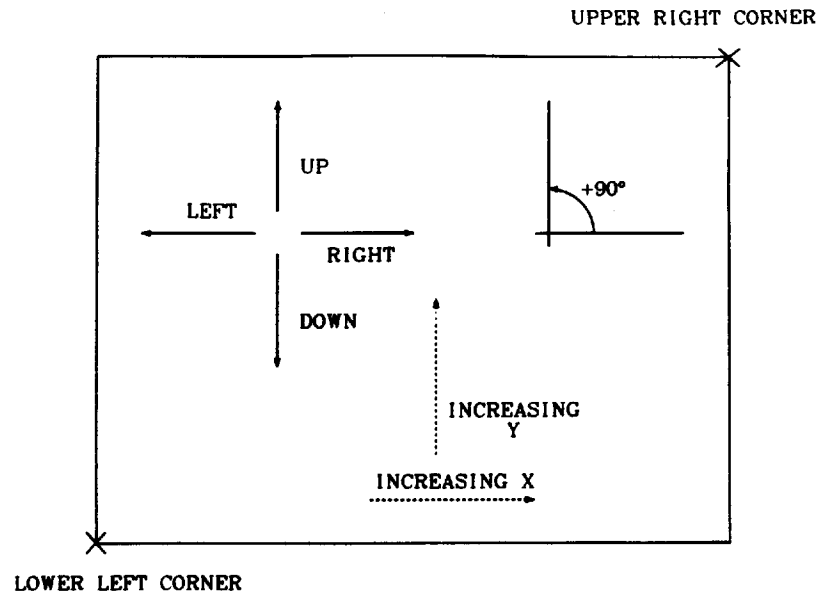
- 1 MAXIMUM VDC EXTENT defines an extent which bounds the VDC extent values which may be found in the metafile. It may be, but need not be, a closest bound in the sense that it exactly equals the union of the extent rectangles in the metafile. This element may be used, for example, to map integer virtual device coordinates of the metafile to a unit square in a normalized device space.

4.4.5 CGM tailoring

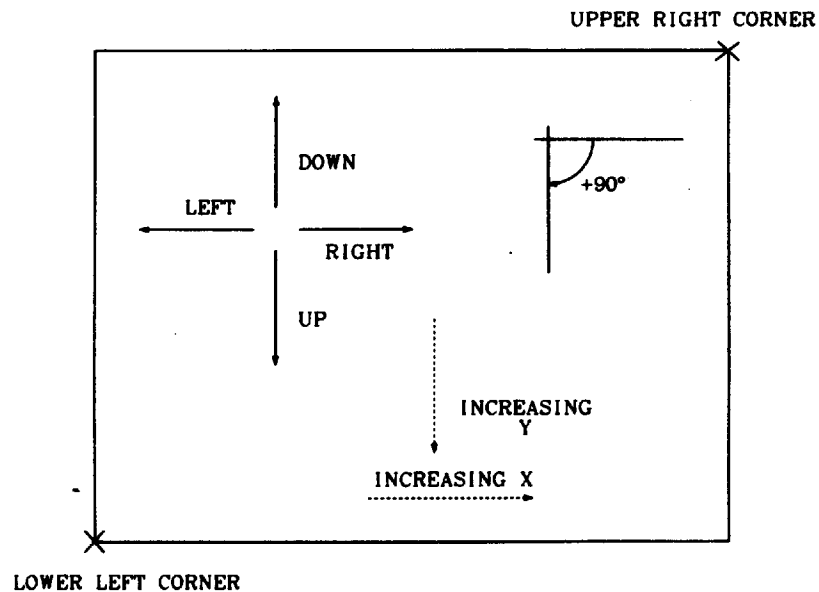
The ability to specify the VDC range and the VDC extent provides the flexibility to configure the metafile addressability in any way desired. It can be configured as an abstract, normalized address range for maximum device independence. It can also be configured to mimic the addressability of a

Concepts

Picture descriptor elements



VDC EXTENT 0.0, 0.0, 1.0, 0.75



VDC EXTENT 0.0, 8.5, 11.0, 0.0

Figure 1 — VDC EXTENT establishes the direction of positive and negative angles.

particular target device in order to take advantage of particular device characteristics. The address range of such a device-specific metafile is just another normalized address range with the normalization limits inherent in the VDC-customizing element; therefore, device independence is maintained.

Picture descriptor elements

Concepts

Such tailoring of the coordinates in a metafile can eliminate the need for transformation of coordinates at metafile interpretation time for the target device. The ability to specify the VDC extent thus allows for the exact registration of coordinates in a metafile with addressable points on the target graphics device.

The use of VDC EXTENT to directly encode world coordinates of large dynamic range and very small granularity will likely result in performance penalties at metafile interpretation time, and may result in decreased portability if such VDC extents exceed those compatible with less capable metafile interpreters.

In addition to VDC tailoring, a metafile generator can limit or tailor the functional content of a metafile to accommodate particular devices or applications, and announce such functional tailoring through the use of METAFILE ELEMENT LIST.

4.4.6 Background colour

Each picture defines a graphical image that is independent of the other images in a metafile. The background colour of the image may be specified by the BACKGROUND COLOUR Picture Descriptor element. If that element is not contained in the Picture Descriptor, the background colour of the image is the default background colour, whether that default is as specified in clause 6 or has been specified in the METAFILE DEFAULTS REPLACEMENT element.

The single parameter of BACKGROUND COLOUR is always RGB, regardless of the current value of COLOUR SELECTION MODE. If the COLOUR SELECTION MODE is indexed, then the BACKGROUND COLOUR element defines the initial representation of colour index 0 for the picture.

1

4.4.7 Device viewport control

The device viewport specifies the region of the device display surface into which the VDC extent is to be mapped on interpretation. VDC-to-Device mapping is determined by the VDC extent, device viewport, and device viewport mapping.

The position of the device viewport is specified in one of three coordinate systems selected by the DEVICE VIEWPORT SPECIFICATION MODE element:

- by fraction [0.0 to 1.0] of the available display surface, which allows reasonable placement and relative sizing of the viewport;
- in millimetres times a scale factor, which allows absolute sizing of images;
- in physical device coordinates.

The device viewport is specified in terms of two points on the device display surface at diagonally opposite corners of the rectangle. Mirroring or 180° rotation of the image may be achieved by specifying the corners in some way other than the first as below and to the left of the second.

The DEVICE VIEWPORT MAPPING element may be used to force isotropic mapping even if the specified VDC extent and device viewport would not otherwise have led to one. In such a case, the VDC extent is mapped on to a subset of the specified device viewport on interpretation. This subset is defined by shrinking either the vertical or horizontal dimension of the specified viewport as needed to reach the required aspect ratio. This smaller "effective viewport" is then used to define the coordinate mapping from VDC to the device's coordinates. The placement of the effective viewport rectangle within the original one can be specified. This placement can be one of 'left', 'right' or 'centred' when the shrinking is horizontal, and 'top', 'bottom' or 'centred' when it is vertical. These meanings are relative to the display surface of the device.

The VDC-to-Device mapping maps the first point specifying the VDC extent on to the corner of the effective viewport corresponding to the first point specifying the device viewport, and similarly for the second point. The mapping is linear in each dimension, but is not necessarily isotropic (for example, a circle in VDC may not appear as a circle to the viewer).

Both the way VDC space is oriented relative to the display surface and the way the effective viewport is placed on the physical device may lead to mirroring and 180° rotation.

The behaviour of primitives and attributes with significance in VDC space under transformations is further described in 4.6.

If both device viewport and scaling mode appear in the same metafile then the last specified is used. If neither appears then the default values for device viewport take precedence.



The elements LINE REPRESENTATION, MARKER REPRESENTATION, TEXT REPRESENTATION, FILL REPRESENTATION and EDGE REPRESENTATION are used to set all of the attribute values in a bundle table entry at the same time. The attributes that may be bundled are described in 4.7.

4.5 Control elements

Control elements specify address space, clipping boundaries, and format descriptions of the CGM elements. Control of some of these format descriptions may be accomplished by Metafile Descriptor elements, while control of others is accomplished by control elements, which may appear in the picture bodies in the metafile. Those items in the former category are fixed for a given metafile, while those in the latter category are changeable; that is, they may change within a picture.



Some of the control elements may appear in the Picture Descriptor if this is permitted by the formal grammar for the metafile version.

4.5.1 VDC space and range

The graphical primitive elements of a metafile define virtual images. The coordinates of these elements (that is, the addresses of points in the virtual image) are absolute two-dimensional Virtual Device Coordinates (VDC). VDC space is a two-dimensional coordinate space of infinite precision and infinite extent. Only a subset of VDC space, the VDC range, is realizable. The VDC range comprises all coordinates representable in the format specified by the declared VDC TYPE and (depending on the type) the VDC INTEGER PRECISION or VDC REAL PRECISION.

The VDC range cannot be set directly; it is completely determined by VDC TYPE and either VDC INTEGER PRECISION or VDC REAL PRECISION elements in the metafile. These elements are controllable, some by dynamic elements in the metafile body and some by static elements in the MD. Note that the VDC range thus defined (a rectangular subregion of the VDC space) does not enclose a continuum of values, but has a distinct granularity. Regardless of the aspect ratio of the VDC range and the granularity within the range, it is implicit that one VDC unit in the x-direction represents the same distance as one VDC unit in the y-direction in VDC space.

4.5.2 Clipping

In order to defer clipping of graphical primitive elements (particularly, expandable elements such as CIRCLE, CIRCULAR ARC 3 POINT, TEXT, etc.) until metafile interpretation time, a clipping control feature is provided in the CGM. Clipping control is achieved by defining CLIP RECTANGLE in VDC space. Whether clipping to the limits of CLIP RECTANGLE actually occurs at metafile interpretation

time is controlled by the CLIP INDICATOR element that sets the mode of the metafile to 'on' or 'off'. The defaults for CLIP RECTANGLE and CLIP INDICATOR are listed in clause 6.



There are three different clipping modes for lines, markers and edges. The required clipping mode is recorded in the metafile with the elements: LINE CLIPPING MODE, MARKER CLIPPING MODE, and EDGE CLIPPING MODE. When the CLIP INDICATOR associated with a graphical primitive is 'on', only those parts of a graphical primitive that are considered inside the effective clipping region are rendered on interpretation. The object clipping modes allow precise specification as to how clipping is applied to primitives on interpretation.

Clipping may be either 'locus', 'shape' or 'locus then shape'. Conceptually, a locus is a mathematical object like a point or line segment, while a shape is an area in 2-dimensional space. Loci are 0-, 1- or 2-dimensional subsets of real-valued 2-space. For markers and text they are points. For lines they are the individual line segments or portions of arcs. The locus of an area is the shape and the boundary. Shapes reflect the realization of geometric attributes and are generally 2-dimensional subsets of real-valued 2-space.

'Locus' clipping is applied for each portion of a graphic object based on its mathematical location and is independent of the area it will occupy after rendering. For example, no portion of a line segment is rendered if the ideal mathematical line lies outside the effective clipping region (even if its line width would carry some portion of the rendering of it into the clipping rectangle); no portion of a marker is rendered if its location lies outside the clipping rectangle.

If 'locus' clipping is used, the rendering is applied to the locus of the graphic object after clipping. The resulting rendered shape areas may therefore extend outside the effective clipping region.

'Shape' clipping is applied after the abstract rendering of shape in device coordinate space. The 2-dimensional point set associated with the graphic object is intersected with the effective clipping region, which has been transformed to device coordinate space.

'Locus then shape' clipping allows the specification that both 'locus' and 'shape' clipping be applied to graphic objects as described above. In this case however, the rendered shape will not extend outside the effective clipping region. A thick line whose locus is outside the clip rectangle will not have any portion visible even if its line width would carry some portion of the rendering inside the clip rectangle.

Figure 1a shows some examples of the effect of the clipping modes.

When a width or size specification mode is 'scaled', the rendering of shape proceeds in device coordinate space after application of the VDC-to-Device mapping.



When a width or size specification mode is 'absolute', the rendering of shape proceeds, conceptually, in VDC space before application of the copy transformation, before application of the segment transformation and before the VDC-to-Device mapping.

Fill and text primitives do not have associated object clipping modes (though the edge of a fill primitive and the boundary edges of a closed figure do). Clipping for fill primitives is always consistent with 'shape' clipping (see 4.6.4.5). For text primitives, the type of clipping is determined by the associated text precision:

For 'string' precision text, clipping proceeds, on a per string basis, in a manner consistent with 'locus' clipping.

For 'character' precision text, clipping proceeds, on a per character basis, in a manner consistent with 'locus' clipping.

For 'stroke' precision text, the clipping always proceeds in a manner consistent with 'shape' clipping.

NOTE - 'shape' clipping for all text precisions is always allowed by this part of ISO/IEC 8632.

Clip rectangles applied to graphical primitive elements within segments may be subject to transformations in VDC space. Intersection of clip rectangles (untransformed or transformed) may result in polygonal clipping boundaries (see 4.12.5).

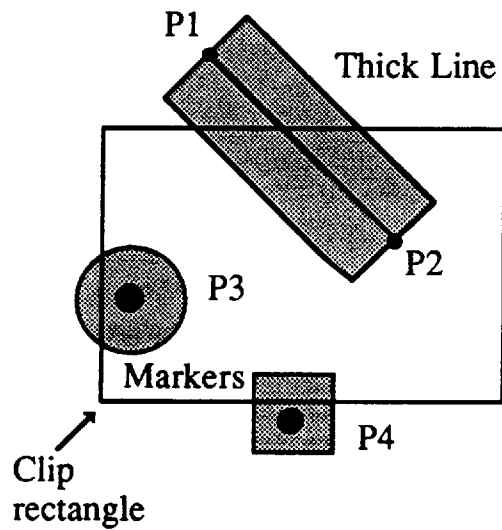


4.5.3 Save and restore primitive context

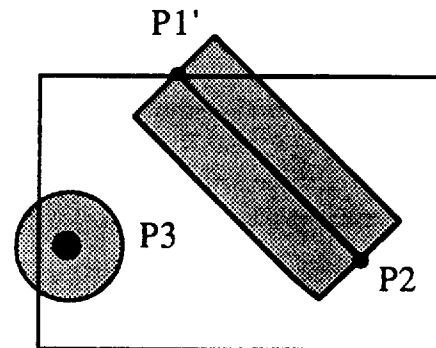
Two elements are provided to save and restore a context; that is, attributes and control elements as collections. This capability allows a list of attributes and control elements (see 5.5.11) to be stored in the metafile which can be referenced by name at a later point in the metafile. This capability can be used to save and restore attributes and control elements in conjunction with opening and closing segments.

The values for attributes controlled by specification or selection modes are saved in the mode in which they were last specified along with the value of the corresponding mode. In restoring a context the current specification and selection modes are not changed.

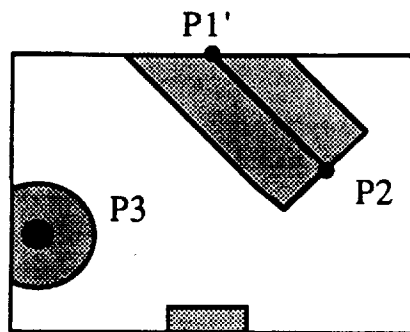
1



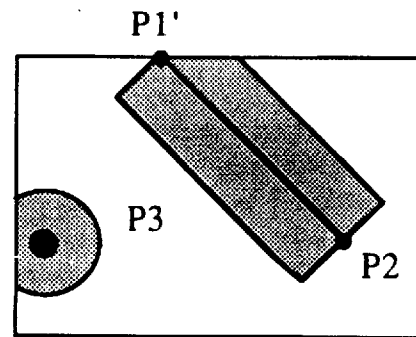
Primitives and clip rectangle stored in the CGM



Picture resulting from 'locus' clipping modes



Picture resulting from 'shape' clipping modes



Picture resulting from 'locus then shape' clipping modes

Figure 1a - Examples of the effects of object clipping modes

4.6 Graphical primitive elements

Graphical primitive elements are those elements that describe the visual components of a picture. Their coordinate arguments are specified in VDC units. The CGM provides the graphical primitive elements

POLYLINE
 DISJOINT POLYLINE
 POLYMARKER
 TEXT
 RESTRICTED TEXT
 APPEND TEXT
 POLYGON
 POLYGON SET
 CELL ARRAY
 GENERALIZED DRAWING PRIMITIVE (GDP)
 RECTANGLE
 CIRCLE
 CIRCULAR ARC 3 POINT
 CIRCULAR ARC 3 POINT CLOSE
 CIRCULAR ARC CENTRE
 CIRCULAR ARC CENTRE CLOSE
 ELLIPSE
 ELLIPTICAL ARC
 ELLIPTICAL ARC CLOSE.
 CIRCULAR ARC CENTRE REVERSED
 CONNECTING EDGE



The metafile supports access to special geometric output capabilities of devices and workstations through the GDP. The GDP has a list of points in VDC as a parameter. It is thus well suited for non-standardized output primitives, which have position, shape, extent, etc., whereas ESCAPE is better suited for non-standardized device control functions.

The formal definition of the CGM describes graphical primitive elements which are positionally independent by virtue of containing complete explicit positional information within each element definition. This "CP-less" model (without the concept of a Current Position) corresponds to that of GKS, and results in fewer side effects and device interaction problems than the CP-oriented graphical models. The efficiency advantages often associated with a CP-oriented model have been realized through techniques used in some of the encodings of ISO 8632.

The TEXT, RESTRICTED TEXT, and APPEND TEXT elements and related text attribute elements are defined in the current VDC space. Thus, they are affected by changes to the Virtual Device Coordinate format.

Five types or categories of graphical primitive elements are defined for the CGM: line elements, marker element, text elements, filled-area elements and cell array element.

The line elements are:

POLYLINE
 DISJOINT POLYLINE
 CIRCULAR ARC 3 POINT
 CIRCULAR ARC CENTRE
 ELLIPTICAL ARC



CIRCULAR ARC CENTRE REVERSED
 CONNECTING EDGE

The marker element is:

POLYMARKER

Graphical primitive elements**Concepts**

The text elements are:

TEXT
RESTRICTED TEXT
APPEND TEXT

The filled-area elements are:

POLYGON
POLYGON SET
RECTANGLE
CIRCLE
CIRCULAR ARC 3 POINT CLOSE
CIRCULAR ARC CENTRE CLOSE
ELLIPSE
ELLIPTICAL ARC CLOSE

The cell array element is:

CELL ARRAY

In addition to these five classes of element, the GENERALIZED DRAWING PRIMITIVE (GDP) is a graphical primitive element that may be used to access device (or implementation) specific graphical primitives that are not accessed by the standardized elements.

In addition to the graphical primitive elements listed above, this part of ISO/IEC 8632 defines elements permitting the definition of 'compound primitives' from several of the other graphical primitives. The following classes of compound primitives are defined: 'compound text' and 'closed figures'. The elements that may be used to specify compound primitives are listed in table 1a.

Table 1a - Contributing primitives to compound primitives

Compound Primitive	First Element	Primitives Included	Other Elements	Final Element
Compound Text	TEXT RESTRICTED TEXT (Note 1)	APPEND TEXT (Note 2) GDP (Note 5)		APPEND TEXT (Note 3) GDP (Note 5)
Closed Figure	BEGIN FIGURE	Line Primitives Fill Primitives (Note 4) GDP (Note 5)	NEW REGION	END FIGURE

NOTES

- 1 The final/not final flag is 'not final'; the primitive defines the reference point of the entire compound text primitive; the text of the primitive is accumulated.
- 2 The final/not final flag is 'not final'.
- 3 The final/not final flag is 'final'; the text of the primitive is accumulated before the compound primitive is closed.
- 4 All primitives of the identified classes may be included.
- 5 Whether a GDP may contribute to compound text or closed figures, and whether or how it specifies that the compound text state or closed figure state be opened, maintained or closed, is specified with the definition of the GDP in the International Register of Graphical Items.

Graphical primitive elements and compound primitive elements may be subject to transformation in VDC space (segment and copy transformation, see 4.12.4.2 and 4.12.5). Such a transformation may change the shape of some primitives. If there is a skew, a primitive initially specified as a rectangle may become a parallelogram. If there is an anisotropic scaling, a primitive initially specified as a circle may become an ellipse. Note that the shape of markers is not affected by such transformations. Anisotropic transformation will change the angle at which non-parallel lines intersect; isotropic transformation will preserve the angle at which non-parallel lines intersect.

4.6.1 Line elements

4.6.1.1 Description

There are two general line elements - POLYLINE and DISJOINT POLYLINE - and three line elements relating to circles and ellipses.

POLYLINE	generates a set of connected lines as defined by a list of points, starting with the first, drawing a line through each successive point, ending at the last point.
DISJOINT POLYLINE	generates a set of unconnected lines as defined by a list of point pairs, drawing from the first to the second, the third to the fourth, etc..
CIRCULAR ARC xxx	generates a single circular arc; two parameterizations of the arc are possible; these are described in 5.6.13 and 5.6.15. A reverse direction arc can also be specified; see 5.6.20.
ELLIPTICAL ARC	generates a single elliptical arc; the parameterization of the arc is described in 5.6.18.



CONNECTING EDGE A line segment connecting the last point of the preceding line element to the next point is generated during the construction of a closed figure. The next point is either the first point of the next line element or the current closure point.

4.6.1.2 Attributes

The appearance of all line elements is controlled by the line attributes, the LINE BUNDLE INDEX and the ASPECT SOURCE FLAGS associated with the line attributes that may be bundled. These are described in 4.7.1.

4.6.1.3 Usage of line elements

POLYLINE is the most general of the primitives. DISJOINT POLYLINE is intended for situations where the alternative would be a large number of "2 point" POLYLINE elements. The ARC primitives provide data compression by comparison with POLYLINE and allow the arcs to be described without knowledge of the resolution of the final viewing surface.



In version 2 metafiles, line clipping is controlled by the LINE CLIPPING MODE element, which can have one of the following values: 'locus', 'shape', or 'locus then shape'. However, clipping applies only if the CLIP INDICATOR is 'on'.

For 'locus' clipping, the mathematical locus of the line is clipped at the intersection with the clip rectangle before shape rendering is applied. Hence, part of the shape of a clipped line may appear outside the clip rectangle.

For 'shape' clipping, the shape of the rendered line is clipped to the intersection with the clip rectangle; that is, nothing is drawn outside the clip rectangle. A portion of a widened line may appear inside the clip rectangle even though the mathematical locus of the line itself may be entirely outside the clip rectangle.

For 'locus then shape' clipping, the mathematical locus of the line is clipped, as with locus clipping, and then subsequently the rendered shape of the clipped locus is again clipped. Note that, since the mathematical locus of the line may have changed as a result of locus clipping, subsequent shape rendering and clipping may produce a different appearance of a line from either of the other two clipping modes.

If the line width is measured in VDC units it is subject to the VDC-to-Device mapping (4.4.7) as well as to both segment and copy transformation (4.12.4.2 and 4.12.5). Note that the entire locus of an arc is subject to these transformations. In the case of an anisotropic mapping or transformation the rendered width of the line will change with the direction of the line segment. If the line width is specified as a scale factor it is not affected by any transformations.

4.6.2 Marker element

Concepts

Graphical primitive elements

4.6.2.1 Description

There is a single marker element.

POLYMARKER generates symbols of a specific type at each of a list of points.

4.6.2.2 Attributes

The appearance of the markers is controlled by the marker attributes, the MARKER BUNDLE INDEX and the ASPECT SOURCE FLAGS associated with the marker attributes that may be bundled. These are described in 4.7.2.

4.6.2.3 Usage of the marker element



The following discussion applies to version 1 metafiles.

Markers conceptually indicate the location of their specifying points. Therefore, if the CLIP INDICATOR is 'on', the marker is visible if, and only if, its specifying point is within the CLIP RECTANGLE. If its specifying point is inside the CLIP RECTANGLE but part of the marker lies outside, the manner in which the marker is clipped (or not) is not standardized for version 1 metafiles.



In situations where there is a requirement that includes making visible those parts of a symbol that are inside the CLIP RECTANGLE while the specifying position of the symbol is outside, the most appropriate primitive element is the TEXT element, used with a single character text argument. For TEXT, three PRECISIONs are specified which control the precision with which each element is clipped. STROKE precision requires clipping even within the body of a symbol. Achieving the centring of the symbol at the specifying position is possible with the TEXT ALIGNMENT element.



In version 2 metafiles, marker clipping is controlled by the MARKER CLIPPING MODE element, which can have one of the following values: 'locus', 'shape' or 'locus then shape'. However, clipping applies only if the CLIP INDICATOR is 'on'.

For 'locus' clipping, the specifying points of each marker are clipped at the intersection with the clip rectangle before shape rendering is applied. The marker is only visible if its specifying point is within the clip rectangle. Hence, part of the shape of a marker may appear outside the clip rectangle providing its specifying point is within the clip rectangle.

For 'shape' clipping, the shape of the rendered marker symbols are clipped to the intersection with the clip rectangle; that is, nothing is drawn outside the clip rectangle. Portions of the marker symbol may appear inside the clip rectangle even if the marker's position is outside.

For 'locus then shape' clipping, the clipping is first applied to the specifying points of each marker, as with 'locus' clipping, and then subsequently the rendered shape of the markers are again clipped.

If the marker size is measured in VDC units, it is subject to the VDC-to-Device mapping (4.4.7) as well as to both segment and copy transformation (4.12.4.2 and 4.12.5). The shape of markers is never affected by transformations; for example, a circle used as a marker type shall always appear as a circle. Only the marker size may be transformed. For this purpose, conceptually, vectors with length equal to the marker size and arbitrary orientations are transformed; the resulting marker size is determined by the orientation of the vector which maximizes the length under the transformation.

If the marker size is specified as a scale factor it is not affected by any transformations.

4.6.3 Text elements

4.6.3.1 Description

Three text elements are provided.

TEXT	generates a text string (or part of a text string) aligned to a particular point.
RESTRICTED TEXT	generates a text string (or part of a text string) that is constrained within a given area.
APPEND TEXT	generates a part of a text string as a part of a text string started with a TEXT or RESTRICTED TEXT element.

4.6.3.2 Attributes

The appearance of all text elements is controlled by the text attributes, the TEXT BUNDLE INDEX and the ASPECT SOURCE FLAGS associated with the text attributes that may be bundled. These are described in 4.7.3.

Changes to the text attributes TEXT FONT INDEX, CHARACTER EXPANSION FACTOR, CHARACTER SPACING, TEXT COLOUR, CHARACTER HEIGHT, CHARACTER SET INDEX, ALTERNATE CHARACTER SET INDEX, and TEXT BUNDLE INDEX, and to the control elements AUXILIARY COLOUR and TRANSPARENCY, are permitted between a non-final text element and its succeeding APPEND TEXT element.

4.6.3.3 Usage of text elements

Each text element has a 'final/not-final' flag. This permits a text string to be started with a TEXT or RESTRICTED TEXT element and continued with one or more APPEND TEXT elements. Only the last element will have its flag set to 'final'. The initial element is always TEXT or RESTRICTED TEXT; subsequent elements may only be APPEND TEXT.

The attributes that may be changed between a related set of text elements are listed in the description of APPEND TEXT (see 5.6.6). These include those that affect font and character set changes, character size and text colour.

Graphical primitive elements**Concepts**

The current setting of TEXT ALIGNMENT is used to align the complete text string assembled from the separate text elements.

Clipping of text strings is described in 4.7.6.

The vectors specified by the CHARACTER ORIENTATION element (4.7.6) are subject to the VDC-to-Device mapping (4.4.7) as well as to both segment and copy transformation (4.12.4.2 and 4.12.5).

4.6.4 Filled-area elements**4.6.4.1 Description**

There are two general fill elements: POLYGON and POLYGON SET. In addition there are six elements that provide data compression and allow areas to be filled accurately without knowledge of the resolution of the final viewing surface.

POLYGON	generates an area and its edge, defined by a list of points; the style of the area is one of 'hollow', 'solid', 'pattern', 'hatch', or 'empty'; the visibility and style of the edge of the area depend on the edge attributes alone.
POLYGON SET	generates a number of areas and their edges, defined by a list of vertex points and vertex flags; the set of styles is the same as for POLYGON; the vertex flags indicate the different polygons in the set; the vertex flags and the edge attributes together control the visibility and style of individual edge segments of each polygon.
RECTANGLE	generates an upright rectangular area; the set of styles is the same as for POLYGON.
CIRCLE	generates a circle; the set of styles is the same as for POLYGON.
CIRCULAR ARC xxx CLOSE	generates a partial circular area; 'pie' and 'chord' style arcs are possible; two parameterization of the arcs are provided; these are described in 5.6.14 and 5.6.16; the set of styles is the same as for POLYGON.
ELLIPSE	generates an ellipse; the parameterization of the ellipse is described in 5.6.17; the set of styles is the same as for POLYGON.
ELLIPTICAL ARC CLOSE	generates a partial elliptical area; 'pie' and 'chord' style arcs are possible; the parameterization is described in 5.6.19; the set of styles is the same as for POLYGON.

4.6.4.2 Attributes

The appearance of all filled-area elements is controlled by the fill attributes, the FILL BUNDLE INDEX, the EDGE BUNDLE INDEX, and the ASPECT SOURCE FLAGS associated with the fill attributes that may be bundled. These are described in 4.7.4.

4.6.4.3 Usage of fill elements

POLYGON provides for the representation of standard irregular areas. RECTANGLE, because it is upright, is a more efficient parameterization of a rectangle than a POLYGON and may be implemented directly in some systems.

The circular and elliptical fill primitives provide an efficient parameterization and allow the areas to be produced accurately without knowledge of the resolution of the final viewing surface.

POLYGON SET allows a related set of polygons to be represented. All attributes of each of the polygons are the same. The specification of the vertex flags allows disjoint polygons (such as both the body and the dot of the letter 'i'), holes (as in a broad ring) and overlapping areas. Accurate rendering of abutting areas of uniform colour, pattern or hatch is possible with the control provided over individual edge visibility.

4.6.4.4 Interior

Concepts

Graphical primitive elements

The interior of a filled-area element is defined as follows. For a given point, create a straight line starting at that point and going to infinity. If the number of intersections between the straight line and the filled area is odd, the point is within the filled area; otherwise it is outside. If the straight line passes a filled-area vertex tangentially, the intersection count is not affected. If a point is within the filled area, it is included in the area to be filled subject to the rules for boundaries and edges (see 4.7.8).

4.6.4.5 Clipping

If parts of a filled-area element are clipped, then the intersection of the interior and the clip boundary becomes part of the boundary of the resulting clipped area for the purposes of display of the boundary for interior style 'hollow'. If the edge is visible, it is not drawn along the new boundary segments created by the clipping of the area.

Edge clipping is controlled by the EDGE CLIPPING MODE element, which has the same enumerations as LINE CLIPPING MODE. Edges are clipped in the same way that lines are clipped; see 4.6.1.3.

4.6.4.6 Transformation

The entire mathematical locus of rectangles, circular and elliptical filled-area elements is subject to the VDC-to-Device mapping (4.4.7), segment transformations (2.12.4.2) and copy transformations (4.12.5). Because anisotropic transformation does not preserve angles between non-parallel lines, rectangles may become parallelograms and circles may become ellipses.

The vectors of the PATTERN SIZE element are subject to all transformations.

The edge widths are treated in exactly the same way as line widths (4.6.1.3).

Under certain conditions the clip rectangle is subject to the copy transformation (4.12.5).

4.6.5 Cell array element

CELL ARRAY

represents a 2-dimensional array of colour values, which cover a rectangle or parallelogram.

The colour values are either direct colour values or indexes into the COLOUR TABLE, according to the current COLOUR SELECTION MODE. The colour values are in the precision declared by a 'local colour precision' parameter of the CELL ARRAY element.

CELL ARRAY is not controlled by any attributes.

4.6.6 Circular arc elements

The CGM provides for two forms of specification of circular arc elements: a centre-radius specification and a 3-point specification. Each has its advantages and disadvantages with respect to numerical accuracy, relationship of defining data to the VDC range, etc.

When choosing which parameterization to use, one should decide where possible numerical inaccuracy would be least disturbing. The 3-point form specifies exact arc endpoints, but might result in inaccurate centre-point calculations, whereas the centre form specifies exact centre point but might result in roundoff errors on the ends of the arc. The 3-point form would thus be more appropriate for smoothly joining an arc to a polyline in a line drawing, whereas the centre form would be more appropriate for pie charts.

4.6.7 Elliptical elements

4.6.7.1 Geometric concepts

Ellipses are specified by Conjugate Diameter Pairs. A Conjugate Diameter Pair (CDP) of an ellipse is a pair D,d of diameters of the ellipse such that a tangent to the ellipse at each endpoint is parallel to the other diameter. The four tangents to the ellipse at the endpoint of the CDP thus form a parallelogram whose sides are bisected by the endpoints of the diameters.

Any CDP of the ellipse remains a CDP across any graphical transformation which transforms an ellipse into an ellipse. This is demonstrated in figure 2 in which the ellipse has been scaled by a factor of two in the y-direction only.

Thus any CDP of a desired ellipse can be used to specify the ellipse. Note that the (mutually perpendicular) major and minor axes of an ellipse and any pair of perpendicular diameters of a circle are CDP's, although they do not necessarily remain perpendicular across a transformation.

Thus to specify an ellipse, all that is needed is three points:

- the centrepoint of the ellipse;
- two CDP endpoints (one endpoint from each diameter).

Graphical primitive elements

Concepts

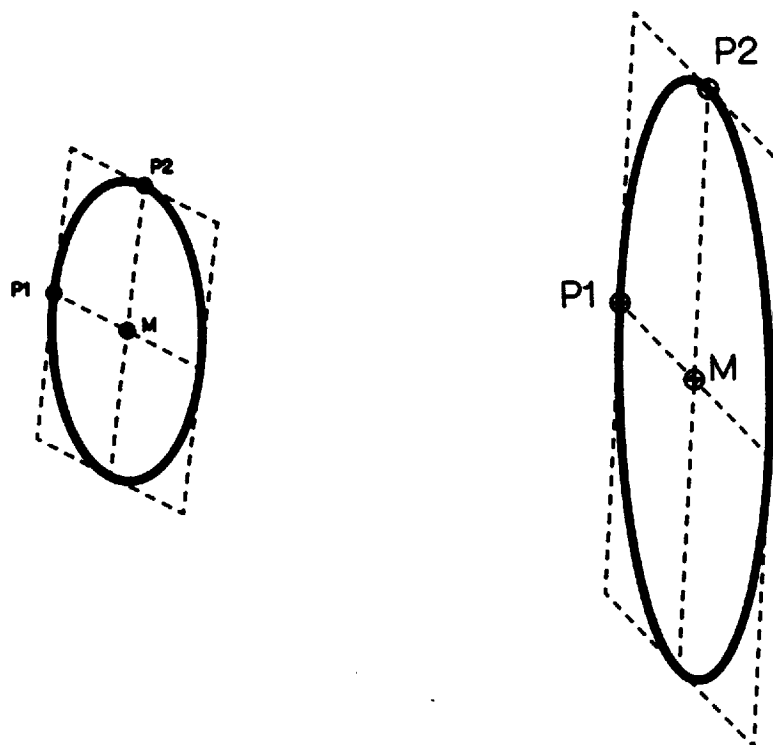


Figure 2 — Anisotropic scaling of an ellipse.

4.6.7.2 Parameterization of elliptical elements in CGM

The ellipse itself in each of the three elliptical elements is parameterized as in the preceding section, the centrepoint and two CDP endpoints. For the two elliptical arc elements, the start and end of the defined arc section is parameterized by two semi-infinite rays originating at the centrepoint. The intersection of these rays with the ellipse defines two points on the ellipse, and these two points define the arc.

4.6.8.1 Construction of closed figures

A closed figure is a fill type compound object which commences with a BEGIN FIGURE element, followed by an ordered sequence of line and fill primitives (and optionally attributes and NEW REGION elements), and followed by END FIGURE. Edge attribute values are associated with the edge portions of the closed figure and fill attribute values are associated with the complete graphic object. BEGIN FIGURE and END FIGURE elements are delimiter elements; NEW REGION is a control element. The entire fill object is considered as a single unit on interpretation.

4.6.8.1.1 Closure point

The first point of the first line primitive in a new region is the closure point for that region. On interpretation this closure point is retained for use in closing the region. When the region is closed (with a NEW REGION or END FIGURE element, or by a fill primitive which begins a new region) an implicit boundary portion from the last point of the last line primitive in the region to this closure point is added to the closed figure on interpretation, unless these points are already coincident.

4.6.8.1.2 Regions

A closed figure consists of one or more regions. A region has a closed boundary which may be concave, convex, or self intersecting. A region is formed either by invoking a fill primitive inbetween BEGIN FIGURE and END FIGURE elements (FIGURE OPEN state; see 4.10) which closes the last region and contributes one or more complete regions, by invoking NEW REGION to start new regions to be formed from line primitives, or by a final invocation of END FIGURE. A closed figure constructed from only line primitives without use of NEW REGION consists of a single region.

The NEW REGION element may occur at any time during the closed figure construction. If the current region is closed, the element is ignored on interpretation. If the current region is open, an implicit boundary portion is added from the last point of the last primitive to the current closure point unless CONNECTING EDGE has been invoked after the last line primitive, in which case, an explicit boundary portion and edge portion is added by the CONNECTING EDGE line primitive.

4.6.8.2 Boundaries and edges

The boundary of each region consists of a combination of implicit boundary portions and edge portions.

4.6.8.2.1 Explicit boundary portions

Explicit boundary portions and edge portions are those added by the inclusion of primitives during closed figure construction. These are generated in the following situations:

- For fill primitives other than POLYGON SET, the complete edge becomes an explicit boundary portion and edge portion in the closed figure.
- For line primitives, those portions which would be rendered outside closed figure construction become explicit boundary portions and edge portions. In particular for DISJOINT POLYLINE, only the segments from the first point to the second point, from the third point to the fourth point, and so on, become explicit boundary portions and edge portions when incorporated into closed figures.
- A CONNECTING EDGE primitive which precedes an action which would normally have added an implicit boundary portion to the closed figure either to close a region (including closing the closed figure itself) or to connect two line primitives results in the portion added being an explicit boundary portion and edge portion. CONNECTING EDGE preceding or following DISJOINT POLYLINE or POLYGON SET does not affect the interpretation of those elements with respect to boundaries and edges.

Edge portions have associated edge attribute values taken from the current attribute values on interpretation. These values can be changed between the line and fill primitives that result in edge portions in a closed figure, and hence each edge portion has a distinct set of attribute values associated with it.

4.6.8.2.2 Implicit boundary portions

Edge attributes are never associated with implicit boundary portions. Implicit boundary portions are only rendered on interpretation for interior style HOLLOW and are a special representation of the interior, not a representation of any portion of the edge.

Implicit boundary portions are added on interpretation to the closed figure definition under the following circumstances:

- When NEW REGION, END FIGURE, or a fill primitive is interpreted and the current region has not been explicitly closed and CONNECTING EDGE has not occurred since the last line primitive, an implicit boundary portion is added from the last point of the last primitive to the current closure point to close the region.
- When the last point of the preceding line primitive is not coincident with the first point of the current line primitive, an implicit boundary portion is created to connect the last point of the preceding line primitive to the first point of the current line primitive.

1

- When portions of a DISJOINT POLYLINE primitive would not normally be rendered (i.e. from the second point to the third point, from the fourth point to the fifth point, and so on), implicit boundary portions are added between these points. (These are additional to the ones which may be added to connect to a preceding or following line primitive or to effect region closure after the disjoint polyline.)
- The portions of a POLYGON SET primitive as described below.

4.6.8.2.3 Conditions under which no boundary or edge is added

No boundary or edge portion is ever created connecting two regions, regardless of how those regions were created or closed.

4.6.8.3 Contribution of primitive elements to the closed figure

4.6.8.3.1 Contribution of line elements to the closed figure

For line primitives, the 'first point' of a line primitive is connected to the 'last point' of the preceding line primitive, and the connecting implicit boundary portion becomes part of the boundary of the closed figure on interpretation. For each of the line primitives the first and last points are defined to be as follows:

POLYLINE p1, p2, ..., pn:
p1 is the first point; pn is the last point.

DISJOINT POLYLINE p1, p2, ..., pn:
p1 is the first point; pn is the last point.

CIRCULAR ARC 3 POINT p1, p2, p3:
p1 is the first point; p3 is the last point.

CIRCULAR ARC CENTRE:
CIRCULAR ARC CENTRE REVERSED:

The first point is the intersection of the circle with the ray (dx start, dy start) from the centre point (i.e. the clockwise end of the arc for CIRCULAR ARC CENTRE, the anti-clockwise end of the arc for CIRCULAR ARC CENTRE REVERSED); the last point is the intersection of the circle with the ray (dx end, dy end) from the centre point (i.e. the anti-clockwise end of the arc for CIRCULAR ARC CENTRE, the clockwise end of the arc for CIRCULAR ARC CENTRE REVERSED).

ELLIPTICAL ARC:

The first point is the intersection of the ellipse with the ray (dx start, dy start) from the centre point; the last point is the intersection of the ellipse with the ray (dx end, dy end) from the centre point.

GENERALIZED DRAWING PRIMITIVE:

For GDPs which generate line primitives, the first point is the first point of the point list; and the last point is the last point of the point list, as defined in the in the GDP registration and associated documentation.

CONNECTING EDGE:

If the region is open, the start point of the connecting edge is the last point of the last line primitive, and the end point of the connecting edge is either the first point of the following primitive or the current closure point as described above. If the connecting edge would be of zero length (i.e. if the two points it connects are coincident), the element is ignored on interpretation. The current modal values of the edge attributes are associated with any edge portion generated by this element.

If the current region is not open, invocations of the CONNECTING EDGE elements encountered are ignored on interpretation (i.e. CONNECTING EDGE shall not be used to connect regions).

Invoking CONNECTING EDGE multiple times after a line primitive results in the first instance (with its associated attributes) being used on interpretation.

On interpretation the theoretical definitions of the line primitives, not their renditions on the display surface, are used to define the explicit boundary portions of the closed figure. In particular, clipping does not apply to the construction of the closed figure, and the gaps or spaces of the edge type or the rendered width of the edge width do not affect the definition of the boundary of the closed figure.

4.6.8.3.2 Contribution of fill elements to the closed figure

Each fill primitive contributes a complete region to the figure (POLYGON SET may contribute more than one), after first closing the current region if one is open. On interpretation, an implicit NEW REGION is performed before and after a fill primitive (i.e. the new region resulting from a fill primitive is closed, and the next primitive begins a new region.)

The unclipped boundary of each fill primitive contributes to the unclipped boundary of the closed figure.



POLYGON SET primitives contribute to closed figure construction as follows:

- A POLYGON SET is considered to contribute one or more complete regions. If the current region has not been closed, an implicit NEW REGION is performed before the POLYGON SET is added to the figure definition. If the POLYGON SET does not end with a point whose edge-out flag is 'close visible' or 'close invisible', an implicit NEW REGION is performed after the POLYGON SET.
- Sequences of points with edge-out flag 'visible' are treated as if they were polylines, terminating with the first point with a different edge-out flag. Each such polyline becomes an edge portion of the boundary of the figure. The edge attribute values (including EDGE VISIBILITY) in effect when POLYGON SET occurs are associated on interpretation with any edge portion added in this way.
- Sequences of points with edge-out flag 'invisible' contribute implicit boundary portions which are polylines joining the points in the sequence, but not edges. Edge attribute values are not associated with these.
- Points with edge-out flag 'close invisible' generate the equivalent of a NEW REGION, generating an implicit boundary portion from this point to the current closure point if these are not coincident, and closing the current region.
- Points with edge-out flag 'close visible' generate the equivalent of a CONNECTING EDGE followed by a NEW REGION, resulting in an edge portion from this point to the current closure point if these are not coincident. The edge attribute values (including EDGE VISIBILITY) in effect when POLYGON SET is invoked are associated with any edge portion added in this way.

4.6.8.3 Contribution of GDPs to the closed figure

A GDP which is defined as a line primitive shall specify which is the first point and the last point in its point list, with respect to closed figure construction. Such GDPs are assumed to contribute to a closed figure a boundary corresponding to the unclipped locus which would be rendered on interpretation if the element occurred outside closed figure construction. Any other behaviour shall be as documented explicitly in the GDP description. A GDP which is defined as being a fill primitive is treated as described in the previous section. Any variation or special handling for closed figure construction shall be documented explicitly in the GDP description.

4.6.8.5 Examples of closed figures

Examples of closed figures are shown in Figure 2a, 2b, 2c, 2d, 2e and 2f.

The POLYGON SET example shown in Figure 13 may also be obtained using the closed figure:

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
  POLYLINE (P3, P1, P2)
  NEW REGION [Note 1]
  POLYLINE (P4, P5, P6, P4)
END FIGURE
```

NOTE

- 1 Invisible implicit boundary portion P3..P2 generated.

Figure 2a shows the closed figure resulting from interpretation of the elements listed below.

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
  POLYLINE (P1, P2)
  CIRCULAR ARC 3 POINT (P2, P3, P4)
  POLYLINE (P4, P5)
  CIRCULAR ARC 3 POINT (P5, P6, P1)
END FIGURE
```

1

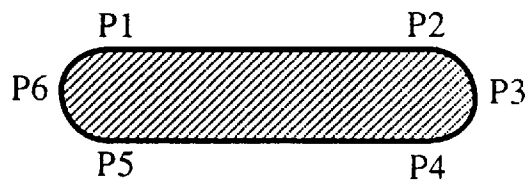


Figure 2a

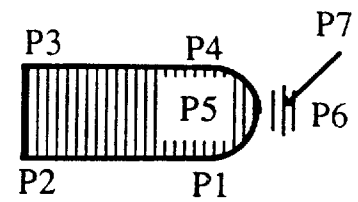


Figure 2b

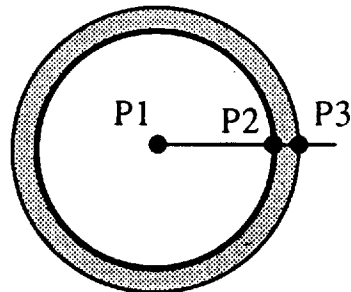


Figure 2c

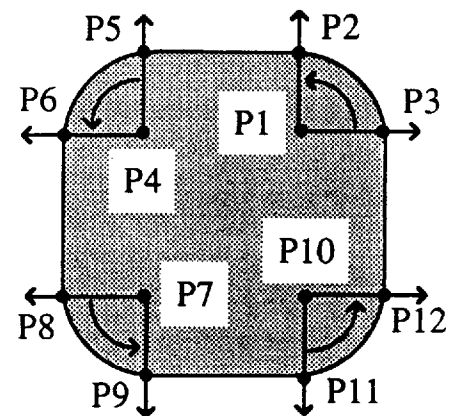


Figure 2d

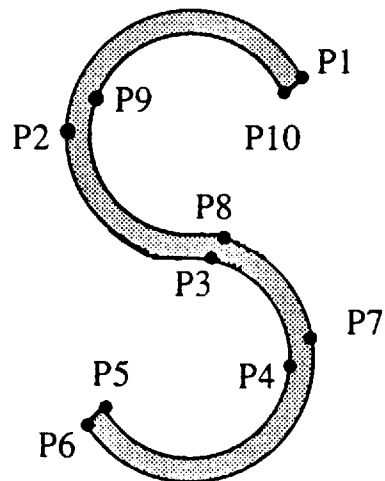


Figure 2e

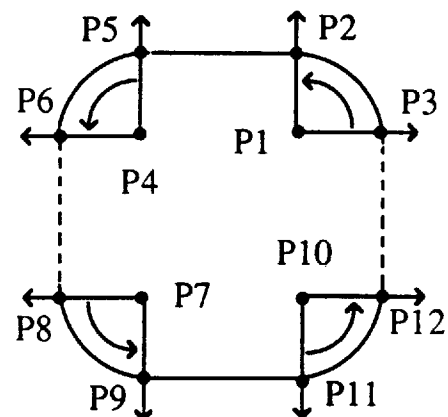


Figure 2f

Figures 2a, 2b, 2c, 2d, 2e, 2f - Examples of closed figures



Figure 2a could also be the result of interpreting the following sequence of elements which include CONNECTING EDGE.

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
  CIRCULAR ARC 3 POINT (P2, P3, P4)
  CONNECTING EDGE
  CIRCULAR ARC 3 POINT (P5, P6, P1) (Note 1)
  CONNECTING EDGE
END FIGURE (Note 2)
```

NOTES

- 1 Visible edge portion P4..P5 generated.
- 2 Visible edge portion P1..P2 generated.

Figure 2b shows the closed figure resulting from interpretation of the elements listed below.

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
  POLYLINE (P1, P2, P3, P4)
  CIRCULAR ARC 3 POINT (P4, P5, P1)
  EDGE VISIBILITY (OFF)
  NEW REGION
  P7 = P5 + (P6-P5)/2
  CIRCULAR ARC CENTRE (P7, 1, 0, 1, 0, |P7 - P5|)
END FIGURE
```

Figure 2c shows the closed figure resulting from interpretation of the elements listed below.

```
BEGIN FIGURE
  CIRCULAR ARC CENTRE (P1, 1, 0, 1, 0, |P3 - P1|)
  NEW REGION
  CIRCULAR CENTRE (P1, 1, 0, 1, 0, |P2 - P1|)
END FIGURE
```

Figure 2c could also be the result of interpreting the following sequence of elements which include fill area elements.

```
BEGIN FIGURE
  CIRCLE (P1, |P3 - P1|)
  CIRCLE (P1, |P2 - P1|)
END FIGURE
```

Figure 2d shows the use of ELLIPTICAL ARC to draw a box with rounded corners and is the result of interpreting the sequence of elements shown below.

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
  ELLIPTICAL ARC (P1, P2, P3, (1,0), (0,1))
  CONNECTING EDGE
  ELLIPTICAL ARC (P4, P5, P6, (0,1), (-1,0)) (Note 1)
  CONNECTING EDGE
  ELLIPTICAL ARC (P7, P8, P9, (-1,0), (0,-1))
  CONNECTING EDGE
  ELLIPTICAL ARC (P10, P11, P12, (0,-1), (1,0))
  CONNECTING EDGE
END FIGURE (Note 2)
```

NOTES

- 1 Visible edge portion P2..P5 generated; edge portions P6..P8 and P9..P11 are drawn with the next two arcs.
- 2 Visible edge portion P12..P3 generated.



Figure 2e shows the use of CIRCULAR ARC 3 POINT to create an 'S' shape and is the result of interpreting the sequence of elements shown below.

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
  CIRCULAR ARC 3 POINT (P1, P2, P3)
  CIRCULAR ARC 3 POINT (P3, P4, P5)
  CONNECTING EDGE
  CIRCULAR ARC 3 POINT (P6, P7, P8) {Note 1}
  CIRCULAR ARC 3 POINT (P8, P9, P10)
  CONNECTING EDGE
END FIGURE {Note 2}
```

NOTES

- 1 Visible edge portion P5..P6 generated.
- 2 Visible edge portion P10.P1 generated.

Figure 2f shows the closed figure resulting from interpretation of the elements listed below. It is similar to figure 2d, but makes use of changing the edge attributes between successive occurrences of CONNECTING EDGE.

```
EDGE VISIBILITY (ON)
BEGIN FIGURE
  ELLIPTICAL ARC(P1, P2, P3, (1,0), (0,1))
  ELLIPTICAL ARC(P4, P5, P6, (0,1), (-1,0)) {Note 1}
  EDGE TYPE (DASHED)
  CONNECTING EDGE
  ELLIPTICAL ARC(P7, P8, P9, (-1,0), (0,-1)) {Note 2}
  EDGE TYPE (SOLID)
  CONNECTING EDGE
  ELLIPTICAL ARC(P10, P11, P12, (0,-1), (1,0))
  EDGE TYPE(DASHED)
  CONNECTING EDGE
END FIGURE {Note 3}
```

NOTES

- 1 No edge portion P2..P5 generated.
- 2 Visible (dashed) edge portion P6..P8 generated; solid edge portion P9..P11 drawn with the next arc.
- 3 Visible (dashed) edge portion P12..P3 generated.

4.7 Attribute elements

Attribute elements determine the appearance of graphical primitive elements. Attributes are classified as either individual attributes or attributes that may be bundled. Table 1 lists the attributes by this classification.

Table 1 — Individual attributes and attributes that may be bundled.

Individual	May Be Bundled
CHARACTER HEIGHT	LINE TYPE
CHARACTER ORIENTATION	LINE WIDTH
TEXT PATH	LINE COLOUR
TEXT ALIGNMENT	MARKER TYPE
CHARACTER SET INDEX	MARKER SIZE
ALTERNATE CHARACTER SET INDEX	MARKER COLOUR
EDGE VISIBILITY	TEXT FONT INDEX
FILL REFERENCE POINT	TEXT PRECISION
PATTERN SIZE	CHARACTER EXPANSION FACTOR
	CHARACTER SPACING
	TEXT COLOUR
	INTERIOR STYLE
	FILL COLOUR
	HATCH INDEX
	PATTERN INDEX
	EDGE TYPE
	EDGE WIDTH
	EDGE COLOUR

Bundled selection of attributes implies that the appearances of graphical primitive elements are distinguishable from one another when different bundles are specified. The method of specification of the aspects that may be bundled of a graphical primitive element may be chosen separately for each aspect. A further group of attributes called ASPECT SOURCE FLAGS (ASFs) takes the values 'individual' and 'bundled' to specify the choice. There is one ASF for each aspect that may be bundled of each primitive.

There is a current modal value for every attribute. Elements are provided to change these modal values. The modal value established by setting an attribute remains until it is explicitly changed. All attributes return to their default values when the BEGIN PICTURE element is encountered.

There is at least one bundle index associated with each of the graphical primitive element types — line, marker, filled area, and text. Line, marker, and text elements have a single associated bundle index. Filled-area elements have two associated bundle indexes, one for interior attributes and one for edge attributes.

The value of each bundle index attribute is modally bound to subsequent graphical primitive elements of the associated type. Distinct values of the bundle index correspond to distinct appearances of the graphical primitive element. For each attribute that may be bundled, there is an associated Aspect Source Flag (ASF).

For individual attributes, the current modal value is used to display a graphical primitive element. For attributes that may be bundled a graphical primitive element is displayed as follows:

- a) if the ASF for an aspect is 'individual', the value used is the current modal value (which is set only by the individual aspect-setting elements);

Attribute elements**Concepts**

- b) if the ASF for an aspect is 'bundled', the value used is obtained via the bundle table for that primitive; the corresponding component of the bundle, which is pointed to by the bundle index, is used.

The actual resulting appearance is interpreter dependent, but the intent is that the interpreter render distinct appearances of graphical primitive elements for distinct values of the associated bundle index (or indexes) by manipulation of the attributes that may be bundled. For example, LINE BUNDLE INDEX designates visually distinct combinations of the polyline attributes LINE WIDTH, LINE TYPE, and LINE COLOUR. Table 2 lists the aspects of each bundle.

Because inquiry of bundle representations is not generally possible in a metafile environment, mixing of 'individual' and 'bundled' ASF values within a bundle will compromise the guarantee of distinguishability of different bundle indexes within that bundle at interpretation time.

Table 2 — Aspects of the bundle and affected primitives.

Bundle	Aspects	Affected primitives
LINE	LINE TYPE LINE WIDTH LINE COLOUR	POLYLINE DISJOINT POLYLINE CIRCULAR ARC 3 POINT CIRCULAR ARC CENTRE ELLIPTICAL ARC
MARKER	MARKER TYPE MARKER SIZE MARKER COLOUR	POLYMARKER
FILL	INTERIOR STYLE FILL COLOUR HATCH INDEX PATTERN INDEX	POLYGON POLYGON SET RECTANGLE CIRCLE CIRCULAR ARC 3 POINT CLOSE CIRCULAR ARC CENTRE CLOSE ELLIPSE ELLIPTICAL ARC CLOSE
EDGE	EDGE TYPE EDGE WIDTH EDGE COLOUR	POLYGON POLYGON SET RECTANGLE CIRCLE CIRCULAR ARC 3 POINT CLOSE CIRCULAR ARC CENTRE CLOSE ELLIPSE ELLIPTICAL ARC CLOSE
TEXT	TEXT FONT INDEX TEXT PRECISION CHARACTER EXPANSION FACTOR CHARACTER SPACING TEXT COLOUR	TEXT RESTRICTED TEXT APPEND TEXT

4.7.1 LINE bundle

The LINE BUNDLE INDEX selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

- a) **LINE TYPE:** determines the type of the line (for example, 'dotted', 'dashed', etc.) with which the polyline is rendered;

Concepts**Attribute elements**

- b) **LINE WIDTH:** determines the width of the line with which the polyline is rendered;
- c) **LINE COLOUR:** determines the colour in which the polyline is drawn.

4.7.2 MARKER bundle

The **MARKER BUNDLE INDEX** selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

- a) **MARKER TYPE:** determines the symbol that is drawn at the marker position (for example, 'dot', 'plus', etc.);
- b) **MARKER SIZE:** determines the size of the marker symbol;
- c) **MARKER COLOUR:** determines the colour in which the marker symbol is drawn.

4.7.3 TEXT bundle

The **TEXT BUNDLE INDEX** selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

- a) **TEXT FONT INDEX:** determines the style of the graphical display of the text characters;
- b) **TEXT PRECISION:** determines the fidelity with which characters need be displayed and positioned;
- c) **CHARACTER EXPANSION FACTOR:** determines the deviation of the character width/height ratio from the ratio established by the font designer;
- d) **CHARACTER SPACING:** determines the amount of blank space added between characters in a string;
- e) **TEXT COLOUR:** determines the colour in which the text characters are drawn.

4.7.4 Filled-area bundles

There are two bundles associated with filled-area elements.

4.7.4.1 FILL bundle

The **FILL bundle** is associated with the interior attributes of filled-area elements. The **FILL BUNDLE INDEX** selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

- a) **INTERIOR STYLE:** determines which of the classes of interior ('hollow', 'solid', 'pattern', 'hatch', or 'empty') is used to draw a filled-area element;
- b) **FILL COLOUR:** determines the colour in which the interior of a filled-area primitive is drawn. This applies only to interior styles 'hollow', 'solid' and 'hatch' (the drawn boundary of a 'hollow' area is considered as part of the representation of the interior);
- c) **HATCH INDEX:** determines which hatch style is used if 'hatch' interior style is selected;
- d) **PATTERN INDEX:** determines which entry in the pattern table is used if 'pattern' interior style is selected.

4.7.4.2 EDGE bundle

The **EDGE bundle** is associated with the edge attributes of filled-area elements. The **EDGE BUNDLE INDEX** selects one entry in a table of bundled attribute values. The following attributes are in this bundle:

- a) **EDGE TYPE:** determines the line type with which the edges are drawn;

Attribute elements

Concepts

- b) **EDGE WIDTH:** determines the width of the edge;
- c) **EDGE COLOUR:** determines the colour in which the edge is drawn.

4.7.5 Specification modes

The CGM provides the mechanism for both 'absolute' and 'scaled' specification of the modal values of the size-related elements **LINE WIDTH**, **MARKER SIZE**, and **EDGE WIDTH**. 'Absolute' specification means that the sizes are given in VDC units. 'Scaled' specification means that the size is specified as a scale factor to be applied at metafile interpretation time to the device-dependent nominal size for the associated primitive.

4.7.6 TEXT attributes

The representation and placement of text characters on a device is controlled by the attribute elements **TEXT FONT INDEX**, **CHARACTER SET INDEX**, **ALTERNATE CHARACTER SET INDEX**, **TEXT PRECISION**, **CHARACTER EXPANSION FACTOR**, **CHARACTER SPACING**, **TEXT COLOUR**, and **CHARACTER HEIGHT** and by the control elements **AUXILIARY COLOUR** and **TRANSPARENCY**. The placement and orientation of text strings is controlled by the attribute elements **CHARACTER ORIENTATION**, **TEXT PATH**, and **TEXT ALIGNMENT**. **TEXT BUNDLE INDEX** is an index into the text bundle table, each entry of which contains values for the attributes that may be bundled. Although the placement and size of text can be precisely specified by the attributes mentioned, the fidelity of rendering depends on the current **TEXT PRECISION**.

The choice of character font (that is, the style of the characters to be displayed) is determined independently of the character set. However, the specified font will only have meaning if it is related to the character set being used. Roman and Gothic are examples of commonly used fonts for Latin-based alphabets.

The attributes in the character representation and placement group (above) and **TEXT BUNDLE INDEX** may be changed within a string. A **TEXT** element or **RESTRICTED TEXT** element is tagged to show it is not complete and provides only the first portion of the string. The **TEXT** element or **RESTRICTED TEXT** element may be followed by the desired text attribute element(s) and then by an **APPEND TEXT** element, which provides the next portion of the string. This may be repeated as often as necessary, with the final **APPEND TEXT** tagged to indicate that the string is complete. Note that a metafile interpreter generally cannot display any of the text until the string is complete because of **TEXT ALIGNMENT** and the way in which attribute changes affect the definition of the text extent rectangle (see below). Text may be displayed before the string is complete only in the following cases:

Concepts

Attribute elements

Table 3 — Cases allowing display of partial text.

Path	Vertical Alignment	Horizontal Alignment
right	normal vertical or baseline	normal horizontal, left, or continuous (0,0)
left	normal vertical or baseline	normal horizontal, right, or continuous (1,0)
down	top, capline, normal vertical, or continuous (0,1)	normal horizontal or centre
up	baseline, bottom, normal vertical, or continuous (0,0)	normal horizontal or centre

There are several methods for inclusion within a string of characters from different character sets. The method used is determined by the CHARACTER CODING ANNOUNCER Metafile Descriptor element. The default or normal technique is to use the CHARACTER SET INDEX element, and restrict the contents of the text strings to printing characters and spaces (format effector control codes such as CR and LF are permitted, but their interpretation is implementation-dependent). Other settings of the CHARACTER CODING ANNOUNCER or use of the ALTERNATE CHARACTER SET INDEX element permit standardized use of 8-bit characters and the SI, SO, and ESC control codes within the text string, in accordance with ISO 2022. The ALTERNATE CHARACTER SET INDEX element is used to select a character set to be used as both the G1 set and the G2 set. The G1 set is used both for 8-bit characters in columns 10 to 15 of the code table, and with the SO control code. The assignment of meaning to the index parameter of both CHARACTER SET INDEX and ALTERNATE CHARACTER SET INDEX is done with the CHARACTER SET LIST Metafile Descriptor element.

Selection of fonts from font tables is done by the TEXT FONT INDEX element. The assignment of meaning to the index values of TEXT FONT INDEX is done with the Metafile Descriptor element FONT LIST.

The font coordinate system is illustrated in figure 3. The character body encloses all of the drawn parts (kerning excepted) of all characters in the font (that is, no descender extends lower than 'bottom', and no accent mark or oversized symbol extends higher than 'top'). The left and right edges of the character body may be defined on a per-character basis to accommodate variable widths, and proportional spacing. It is expected that font designers will specify some fonts having kerns extending beyond the character body. The body exceeds the actual character symbol width and height as necessary to provide adequate white space between characters, such that text is readable and adequately separated when adjacent character bodies are flush (that is, when CHARACTER SPACING is 0). The character body is defined in this way to permit alignment of multiline text without overlaps in the metafile environment. The CHARACTER HEIGHT specifies the VDC distance between the capline and baseline of the font (see figure 3). The CHARACTER EXPANSION FACTOR specifies the deviation of the width to height ratio of the characters from the ratio indicated by the font designer (see figure 4). CHARACTER SPACING specifies how much additional space is to be inserted between two adjacent character bodies (see figure 5). If the value of CHARACTER SPACING is zero, the character bodies are arranged one after the other along the TEXT PATH with only the intercharacter spacing designated by the font designer. If the value of CHARACTER SPACING is positive, additional space is inserted between character bodies. If the value of CHARACTER SPACING is negative, adjacent character bodies overlap although the character symbols themselves might not. Character spacing is specified as a fraction of the CHARACTER HEIGHT.

Attribute elements

Concepts

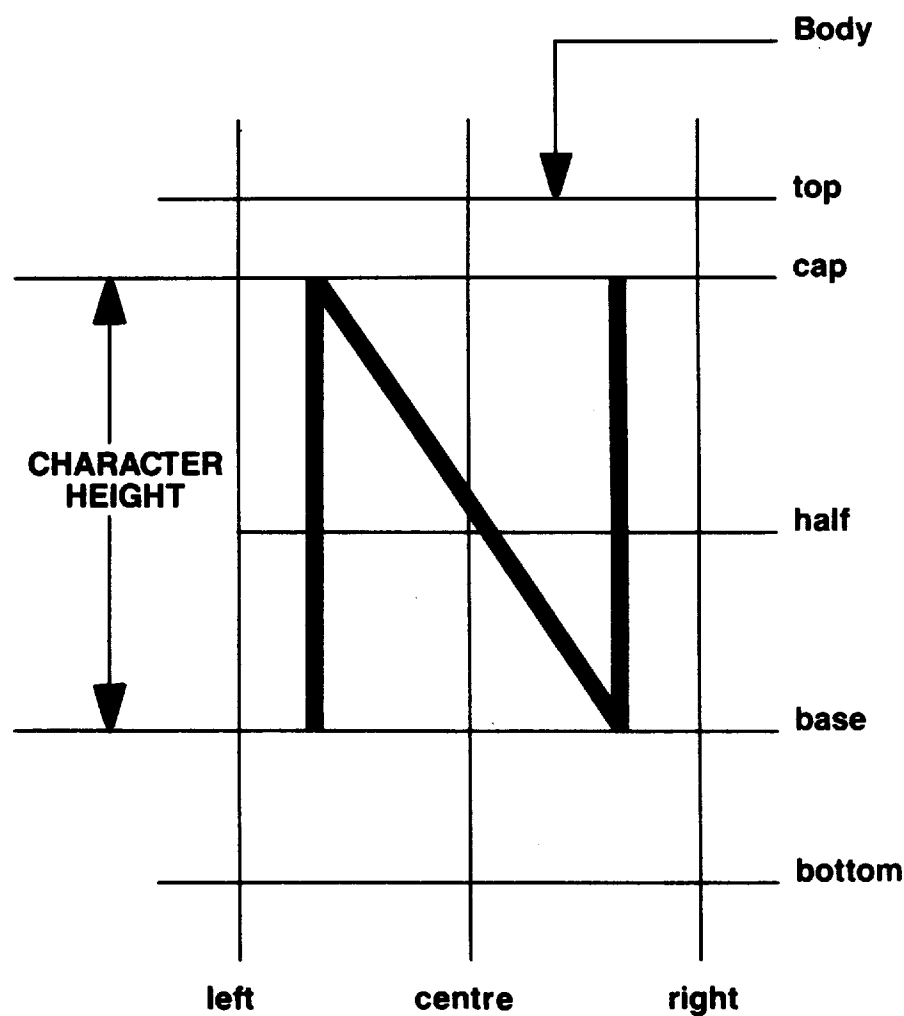


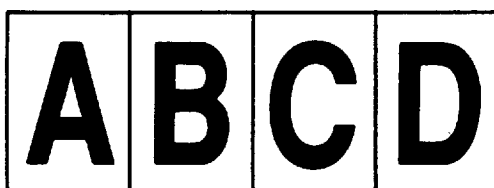
Figure 3 — Font description coordinate system.

Concepts

Attribute elements



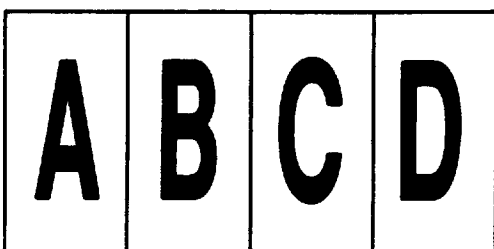
CHARACTER HEIGHT = 1.0
CHARACTER EXPANSION FACTOR = 1.0



CHARACTER HEIGHT = 1.5
CHARACTER EXPANSION FACTOR = 1.0



CHARACTER HEIGHT = 1.0
CHARACTER EXPANSION FACTOR = 1.5



CHARACTER HEIGHT = 2.0
CHARACTER EXPANSION FACTOR = 0.75

Figure 4 — CHARACTER HEIGHT and CHARACTER EXPANSION FACTOR.

Attribute elements

Concepts

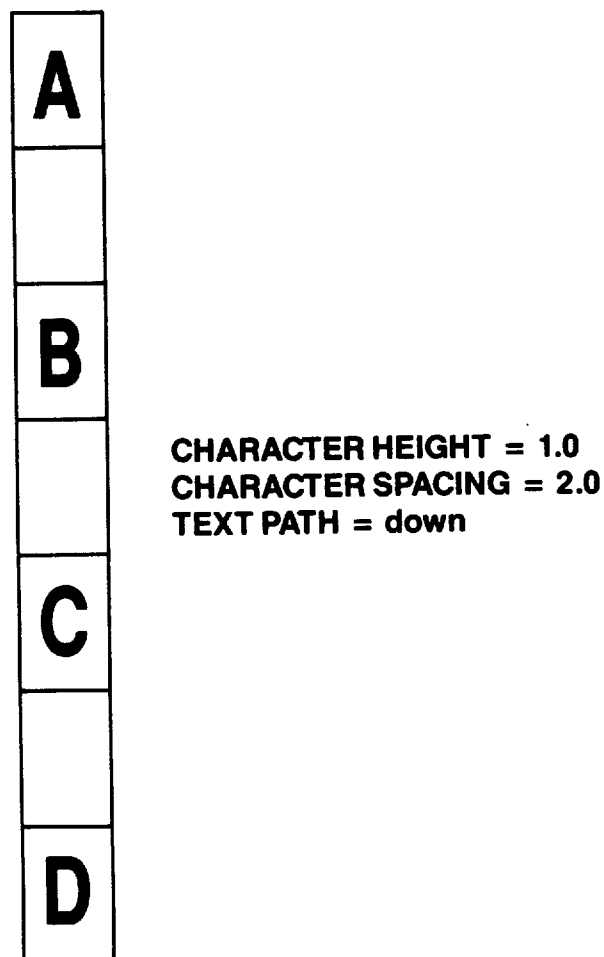
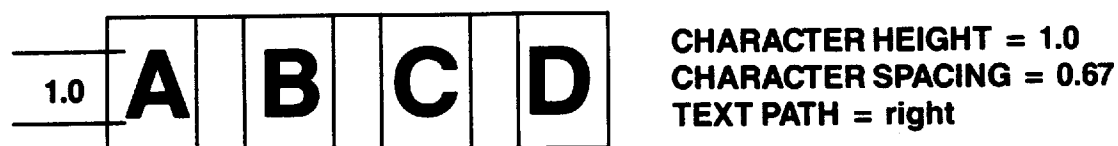
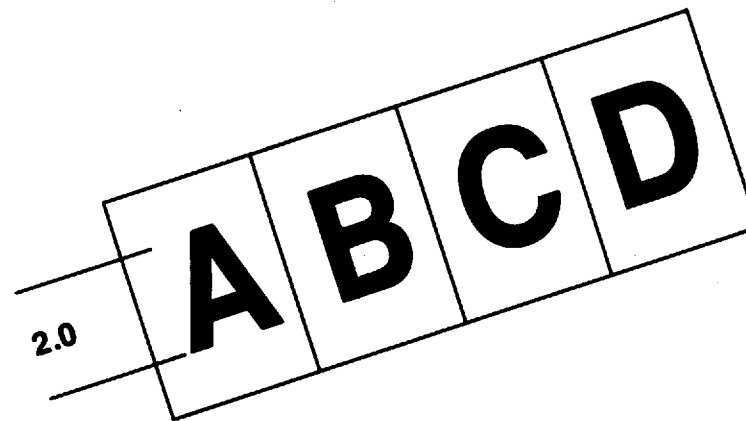


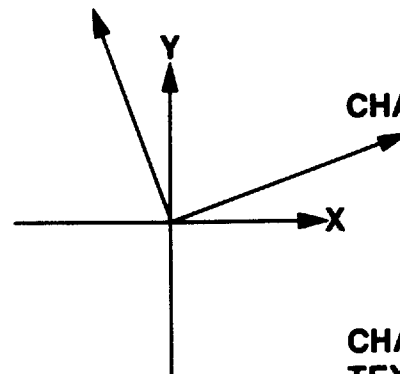
Figure 5 — CHARACTER SPACING.

Concepts

Attribute elements



CHARACTER UP VECTOR = $(-1, 3)$



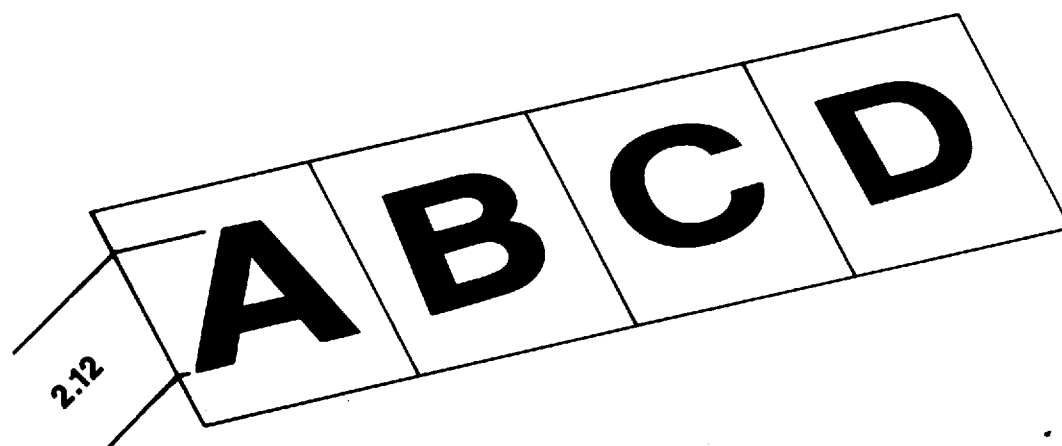
CHARACTER BASE VECTOR = $(3, 1)$

CHARACTER ORIENTATION = $(-1, 3, 3, 1)$
 TEXT PATH = right
 CHARACTER HEIGHT = 2.0

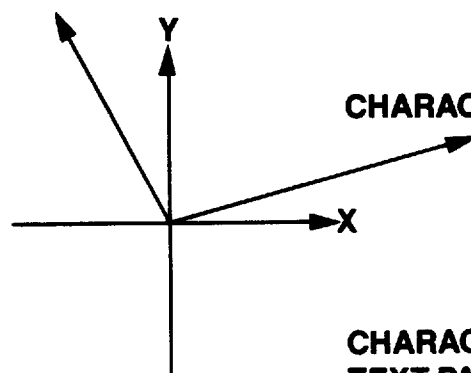
Figure 6 — CHARACTER ORIENTATION.

Attribute elements

Concepts



CHARACTER UP VECTOR = $(-1.5, 3)$



CHARACTER BASE VECTOR = $(4.5, 1)$

CHARACTER ORIENTATION = $(-1.5, 3, 4.5, 1)$

TEXT PATH = right

CHARACTER HEIGHT = 2.12

Figure 7 — CHARACTER HEIGHT and CHARACTER ORIENTATION after anisotropic transformation.

Concepts**Attribute elements**

CHARACTER ORIENTATION specifies the character up vector and base vector, which fix the orientation, skew, and distortion of the characters, and also determine the sense of 'right', 'left', 'up', and 'down' for TEXT PATH and TEXT ALIGNMENT (see figure 6).

The way in which software above the metafile generator and/or the metafile generator itself may use CHARACTER ORIENTATION is described. To generate the CHARACTER ORIENTATION and CHARACTER HEIGHT elements, a vector whose length is the character height (baseline-to-capline) and whose direction is the desired character up vector is created. A second vector is also created with the same length, whose direction is negative 90° from the up vector. This pair of vectors may be transformed before being given to the metafile generator as the parameters to CHARACTER ORIENTATION. The length of the transformed up vector may then be used to generate the CHARACTER HEIGHT element. If an anisotropic transformation is in effect above the metafile generator, the character height must be respecified by the metafile generator for each change in orientation (see figure 7). The CHARACTER HEIGHT and CHARACTER ORIENTATION are decoupled to permit changing character height (but not orientation) within a string. Thus, to the metafile interpreter, the absolute lengths of the vectors in CHARACTER ORIENTATION are not significant; only their directions and the ratio of their lengths are significant.

The ratio of the length of the width vector to the length of the height vector is used to scale the CHARACTER SPACING for text paths 'right' and 'left', and the CHARACTER EXPANSION FACTOR in all cases, before these are used to display the text.

TEXT PATH has the possible values 'right', 'left', 'up', and 'down'. It specifies the writing direction of the text string as follows:

- right: means the direction of the character base vector;
- left: means 180° from the character base vector;
- up: means the direction of the character up vector;
- down: means 180° from the character up vector.

For the 'up' and 'down' text path directions, the characters are arranged so that the centres of the character bodies are on a straight line in the direction of the up vector of CHARACTER ORIENTATION. For the 'left' and 'right' text path directions, the characters are arranged so that the baselines of the characters are on a straight line parallel to the direction of the character base vector. These composition rules also hold true when characters of different heights, expansion factors, fonts, or precisions are intermixed in a string by means of attribute changes between non-final TEXT elements and subsequent APPEND TEXT elements.

Alignment of text is done with respect to a text extent rectangle, which is derived by joining the character bodies of the characters in the string according to the current status of the attributes and the composition rules described. Alignment is performed according to the highest precision in the string.

For TEXT PATH = 'left' or 'right',

- | | |
|-------------|--|
| TOPLINE: | topline farthest from the baseline |
| CAPLINE: | capline farthest from the baseline |
| HALFLINE: | halfline farthest from the baseline |
| BOTTOMLINE: | bottomline farthest from the baseline |
| LEFT: | leftmost edge of leftmost character body |
| RIGHT: | rightmost edge of rightmost character body |
| CENTRE: | halfway between left and right edges |

For TEXT PATH = 'up' or 'down',

- | | |
|-------------|---|
| TOPLINE: | topline of topmost character |
| CAPLINE: | capline of topmost character |
| HALFLINE: | halfway between halflines of topmost and bottommost character |
| BASELINE: | baseline of bottommost character |
| BOTTOMLINE: | bottomline of bottommost character |

