महाराष्ट्र राज्य तंत्रशिक्षण मंडळ, मुंबई

(स्वायत्त) (ISO 9001:2015) (ISO/IEC 27001:2013)

अभियांत्रिकी आणि तंत्रज्ञान पदविका

शिक्षण पुस्तिका

(Learning Material)

COMPUTER GRAPHICS

(22318)

संगणक अभियांत्रिकी गट

मराठी-इंग्रजी (द्विभाषिक) माध्यम

(अभियांत्रिकी व तंत्रज्ञानातील तृतीय सत्र पदविका)

शिक्षण पुस्तिका
(Learning Material)

COMPUTER GRAPHICS

(22318)

संगणक अभियांत्रिकी गट

मराठी-इंग्रजी (द्विभाषिक) माध्यम
(अभियांत्रिकी व तंत्रज्ञानातील तृतीय सत्र पदविका)

महाराष्ट्र राज्य तंत्रशिक्षण मंडळ, मुंबई
(स्वायत्त) (ISO 9001:2015) (ISO/IEC 27001:2013)

मार्गदर्शक

हंगे ज्योती रामराव

विभागप्रमुख, संगणक अभियांत्रिकी

लेखक

वाकचौरे सुष्मा लक्ष्मण

अधिव्याख्याता, संगणक तंत्रज्ञान

बो-हाडे सुप्रिया सुरेंद्र

अधिव्याख्याता, संगणक तंत्रज्ञान

बहेती सपना सचिन

अधिव्याख्याता, संगणक तंत्रज्ञान

शिंदे बिपिन बाळु

प्रभारी विभागप्रमुख, संगणक अभियांत्रिकी

महाराष्ट्र राज्य तंत्र शिक्षण मंडळ

(स्वायत्त) (ISO ९००१:२०१५) (ISO/IEC २७००१:२०१३)

शासकीय तंत्रनिकेतन इमारत, ४ था मजला, ४९, खेरवाडी, वांद्रे (पूर्व), मुंबई - ४०० ०५१

दू.क्र.: ०२२-६२५४२१००/१०१/१०२

संकेतस्थळ : www.msbte.org.in ई-मेल : director@msbte.com

प्रास्ताविक

महाराष्ट्र राज्यातील पदविका स्तरावरील तंत्रशिक्षणाशी संबंधित बाबींचे नियमन करण्यासाठी महाराष्ट्र राज्य तंत्रशिक्षण मंडळ वचनबद्ध असून विद्यार्थ्यांच्या सर्वांगिण विकासाकरिता वेळोवेळी प्रयत्नशील आहे. तंत्रज्ञान, उद्योग, समाज आणि जागतिकीकरण यामध्ये सतत घडून येणा-या बदलांच्या अनुषंगाने तांत्रिक शिक्षणाची भविष्यातील निकड वेधून पदविका स्तरावरील अभ्यासक्रम, परीक्षा पद्धती व शैक्षणिक सामुग्री ह्यांमधे अद्ययावत बदल करण्यात महाराष्ट्र राज्य तंत्रशिक्षण मंडळ अग्रगण्य आहे. विद्यार्थी हा शिक्षण क्षेत्राच्या केंद्रस्थानी असून त्यांची निकड व समस्या संवेदनशीलपणे हाताळल्यास भारत देशाचे 'ज्ञान महासत्ता' बनण्याचे स्वप्न पूर्णत्वास जाईल ह्याचा मला विश्वास आहे.

शहर आणि ग्रामीण भागातील शैक्षणिक सोयीसुविधांमधील दरी अनेक वेळा दिसून येत असून ग्रामीण भागातील विद्यार्थ्यांचे इंग्रजी भाषेतील ज्ञान व संवाद कौशल्याबाबतही ही वस्तुस्थिती प्रकर्षाने जाणवते. केवळ इंग्रजी भाषेतील संवाद कौशल्याअभावी ग्रामीण भागातील विद्यार्थी तंत्रशिक्षणापासून वंचित राहू नये, ह्या दृष्टिकोनातून महाराष्ट्र राज्य तंत्रशिक्षण मंडळाने शैक्षणिक वर्ष २०२१-२२ पासून प्रथम वर्ष पदविका अभ्यासक्रमाकरिता तांत्रिक शिक्षण मराठी-इंग्रजी द्विभाषिक माध्यमात इच्छुक विद्यार्थ्यांना उपलब्ध करून दिले आहे. मात्र असे करताना कोणत्याही परिस्थितीत गुणवत्तेशी तडजोड केली जाऊ नये ह्या दृष्टीने प्रमुख विषयांसाठीच्या शैक्षणिक सामुग्रीची निर्मिती करण्यात आली आहे.

राष्ट्रीय शिक्षण धोरण २०२० मध्ये प्रादेशिक भाषांमध्ये सर्वांना शिक्षणाची कल्पना मांडण्यात आली आहे. त्यास अनुसरून मराठी-इंग्रजी द्विभाषिक माध्यमाचा पर्याय द्वितीय व तृतीय वर्षाकरिताही उपलब्ध करून देण्यात आला आहे. तसेच त्याकरिता शैक्षणिक सामुग्रीही विद्यार्थी व अध्यापकांना उपलब्ध करून देण्यात येत आहे.

महाराष्ट्र राज्यातील अनुभवी अध्यापकांकरवी ही शैक्षणिक सामुग्री तयार करण्यात आली असून व्यावहारिक मराठी भाषा, इंग्रजी भाषेतील तांत्रिक शब्दावलीचा उपयोग आणि संदर्भ पुस्तके लक्षात घेऊन या सामुग्रीची निर्मिती करण्यात आलेली आहे. सदर सामुग्रीची पुनर्तपासणी सुकाणू समितीमार्फत करण्यात आलेली असल्याने ही शैक्षणिक सामुग्री अधिक समृद्ध झालेली आहे. त्यामुळे विद्यार्थ्यांना तांत्रिक शिक्षण समजून घेणे अधिक सुकर होईल. तसेच व्यावहारिक मराठी भाषेच्या उपयोगाने विद्यार्थ्यांना विषयाचे सखोल आकलन होईल व इंग्रजी भाषेतील तांत्रिक शब्दावलीच्या वापरामुळे विद्यार्थ्यांचा उद्योग जगतातील वावर सुलभ होईल. त्यामुळे महाराष्ट्र राज्य तांत्रिक क्षेत्रातील वैश्विक मनुष्यबळाच्या निर्मितीत अग्रेसर राहील व त्यायोगे राष्ट्रनिर्मितीकरीता निश्चितच हातभार लागेल असा मला विश्वास आहे.

अभियांत्रिकी पदविका अभ्यासक्रमातील प्रमुख विषयांची मराठी-इंग्रजी द्विभाषिक शैक्षणिक सामुग्री बनविण्यासाठी अध्यापक व सुकाणू समितीचे सदस्य हे कौतुकास पात्र असून मी त्यांचे अभिनंदन करतो.

विनोद म. मोहितकर

(डॉ. विनोद म. मोहितकर)

संचालक,

महाराष्ट्र राज्य तंत्रशिक्षण मंडळ, मुंबई

अनुक्रमणिका

अ.क्र.	घटकाचे नाव	पान.क्र.
1	घटक 1 संगणक ग्राफिक्स ची बेसिक संकल्पना	1-28
2	घटक 2 रास्टर स्कॅन ग्राफिक्स (Raster Scan Graphics)	29-52
3	घटक 3 परिवर्तनाचा आढावा	53-82
4	घटक 4 विंडोइंग विन्डोइंग आणि क्लिपिंग (Windowing and Clipping)	83-110
5	घटक 5 वक्ररेषेचा परिचय	111-123

घटक १ (Unit I)

संगणक ग्राफिक्स ची बेसिक संकल्पना

(Basics of Computer Graphics) (Marks 08)

विषय निष्पत्ती (Course Outcome): प्रतिमांची व्हिज्युअल आणि भौमितिक माहिती हाताळा.

घटक निष्पत्ती (Unit Outcome):

१ .दिलेल्या मोडचे गुणधर्म वेगळे करा

२. दिलेल्या स्कॅन डिस्प्लेच्या वैशिष्ट्यांची तुलना करा.

३. "C" वापरून दिलेले प्रकार काढण्यासाठी प्रोग्राम लिहा.

४. दिलेल्या डिस्प्ले उपकरणांच्या अनुप्रयोगाचे वर्णन करा.

५. दिलेल्या 2D को-ऑर्डिनेट्सची भौतिक उपकरण समन्वयांशी तुलना करा.

१.१ परिचय करण्यासाठी संगणक ग्राफिक्स

- संगणक ग्राफिक्स म्हणजे संगणकाच्या मदतीने आकृती ,चित्रे ,इमेजेस ,आकार तयार करणे,त्या आकृतीला ऍनिमेशन देणे .
- संगणक ग्राफिक्स ही चित्रे, रेषा, तक्ते इत्यादी काढण्याची एक कला आहे. प्रोग्रामिंगच्या मदतीने संगणक वापरून.
- ही प्रतिमा तयार करणे, हाताळणे, संग्रहित करणे आणि प्रदर्शित करणे अशी प्रक्रिया आहे.
- संगणक ग्राफिक्स दोन वेगवेगळ्या श्रेणींमध्ये विभागले जाऊ शकतात: रास्टर ग्राफिक्स आणि वेक्टर ग्राफिक्स. मूलत: दोन्ही समान उद्दिष्ट (उच्च-गुणवत्तेची डिजिटल प्रतिमा) साध्य करण्यासाठी तयार असताना, ते भिन्न तंत्रे वापरतात. संगणक ग्राफिक्स मध्ये माहिती प्रदर्शित करते . साधी ग्राफिक्स व्याख्या खालीलप्रमाणे आहे: प्रतिमांच्या स्वरूपात अर्थपूर्ण माहिती तयार करण्यासाठी आणि हाताळण्यासाठी संगणकाचा वापर. या प्रतिमा अनेक वेगवेगळ्या स्वरूपात येतात, जसे की आयकॉन आणि स्केचेस. अशा प्रकारे, संगणक ग्राफिक्स सचित्र स्वरूपात डेटा व्यक्त करतात.
- कॉम्प्युटर ग्राफिक्समध्ये पिक्सेलचे संकलन म्हणून ऑब्जेक्ट्सचे प्रतिनिधित्व केले जाते, जेथे पिक्सेल हा सर्वात लहान ॲड्रेस करण्यायोग्य बिंदू आहे जो स्क्रीनवर प्रदर्शित केला जाऊ शकतो. पिक्सेल त्याची तीव्रता आणि रंग सेट करून स्क्रीनवर प्रदर्शित केला जाऊ शकतो.

१.१.१ संगणक ग्राफिक्स ची मुख उपक्षेत्रे:

१. भूमिती: प्रक्रिया पृष्ठभागचा अभ्यास

२. ॲनिमेशन: अनिमेशन स्थिर प्रतिमांचा क्रम प्रदर्शित करून तयार केलेल्या डिस्प्ले डिव्हाइसच्या स्क्रीनवरील हालचालींचा संदर्भ देते. अनिमेशन हे मल्टीमीडिया आणि गेमिंग उत्पादनांमध्ये समाकलित केलेल्या डिझाइनिंग, रेखाचित्रे, लेआउट आणि फोटोग्राफिक मालिका तयार करण्याचे तंत्र आहे

३.प्रस्तुतीकरण: प्रस्तुतीकरण किंवा प्रतिमा संश्लेषण ही संगणक प्रोग्रामद्वारे 2**D** किंवा 3**D** मॉडेलमधून फोटोरिअलिस्टिक किंवा नॉन-फोटोरिअलिस्टिक प्रतिमा तयार करण्याची प्रक्रिया आहे.

४.इमेजिंग: अभ्यास प्रतिमा संपादन किंवा प्रतिमा संपादन

१.२ बेसिक संकल्पना

१.२.१प्रतिमा

- प्रतिमा म्हणजे पिक्सेलचे संकलन.*मुळात प्रतिमा* ही संगणकावरील वास्तविक जगाच्या वस्तूचे प्रतिनिधित्व करते. हे वास्तविक चित्र प्रदर्शन, व्हिडिओ मेमरीमध्ये संग्रहित पृष्ठ किंवा प्रोग्रामद्वारे व्युत्पन्न केलेला कोड आहे.
- प्रतिमा एखाद्या गोष्टीचे दृश्य प्रतिनिधित्व असते, तर डिजिटल प्रतिमा दृश्य डेटाचे बायनरी प्रतिनिधित्व असते. या प्रतिमा छायाचित्रे, ग्राफिक्स आणि वैयक्तिक व्हिडिओ फ्रेम्सचे रूप घेऊ शकतात. या उद्देशासाठी, प्रतिमा एक चित्र आहे जी इलेक्ट्रॉनिक स्वरूपात तयार केली गेली किंवा कॉपी केली गेली आणि संग्रहित केली गेली.

१.२.२ पिक्सेल आणि रिझोल्यूशन

पिक्सेल

- **Pixel** चा अर्थ काय**?**

पिक्सेल हे डिजिटल इमेज किंवा ग्राफिकचे सर्वात लहान एकक आहे जे डिजिटल डिस्प्ले डिव्हाइसवर प्रदर्शित आणि प्रस्तुत केले जाऊ शकते.पिक्सेल हे डिजिटल ग्राफिक्समधील मूलभूत तार्किक एकक आहे. संपूर्ण प्रतिमा, व्हिडिओ, मजकूर किंवा कॉम्प्युटर डिस्प्लेवरील कोणतीही दृश्यमान वस्तू तयार करण्यासाठी पिक्सेल एकत्र केले जातात.**पिक्सेल हा सर्वात लहान संबोधित करण्यायोग्य स्क्रीन घटक आहे.**

पिक्सेलला सर्वात लहान आकाराचे ऑब्जेक्ट किंवा रंग स्पॉट म्हणून परिभाषित केले जाऊ शकते जे मॉनिटरवर प्रदर्शित आणि संबोधित केले जाऊ शकते.पिक्सेल उभा आहे च्या साठी चित्र घटक. ए पिक्सेल आहे सर्वात लहान तुकडा च्या माहिती एका प्रतिमेत.कॉम्प्युटर ग्राफिक्समध्ये पिक्सेलचा संग्रह म्हणून ऑब्जेक्ट्सचे प्रतिनिधित्व केले जाते, जेथे पिक्सेल आहे सर्वात लहान ॲड्रेस करण्यायोग्य बिंदू जो स्क्रीनवर प्रदर्शित केला जाऊ शकतो.

- **पिक्सेल रिझोल्यूशन**

पिक्सेल रिझोल्यूशनमध्ये, रिझोल्यूशन हा शब्द डिजिटल प्रतिमेतील पिक्सेलच्या एकूण संख्येचा संदर्भ देतो. उदाहरणार्थ. जर एखाद्या प्रतिमेमध्ये **M** पंक्ती आणि **N** स्तंभ असतील, तर त्याचे रिझोल्यूशन **M X N** म्हणून परिभाषित केले जाऊ शकते. जर आपण पिक्सेलची एकूण संख्या म्हणून रिझोल्यूशन परिभाषित केले, तर पिक्सेल रिझोल्यूशन दोन संख्यांच्या संचासह परिभाषित केले जाऊ शकते.

पिक्सेल रिझोल्यूशनमध्ये, रिझोल्यूशन हा शब्द एकूण क्र. डिजिटल प्रतिमेमध्ये पिक्सेलची संख्या.

उदाहरणार्थ, जर एखाद्यामध्ये **M** पंक्ती आणि **N** स्तंभ असतील, तर त्याचे रिझोल्यूशन **MX N** म्हणून परिभाषित केले जाऊ शकते.

फ्रेमबफर (फ्रेम बफर, किंवा कधीकधी फ्रेमस्टोअर) हा यादृच्छिक-प्रवेश मेमरी (**RAM)** चा एक भाग असतो ज्यामध्ये बिटमॅप असतो जो व्हिडिओ डिस्प्ले चालवतो. हा एक मेमरी बफर आहे ज्यामध्ये संपूर्ण व्हिडिओ फ्रेममधील सर्व पिक्सेल दर्शविणारा डेटा आहे. आधुनिक व्हिडिओ कार्ड्समध्ये त्यांच्या कोरमध्ये फ्रेमबफर सर्किटरी असते.

१.२.३ रास्टरायझेशन :

रास्टरायझेशनमध्ये, वेक्टर ग्राफिक्स फॉरमॅट (आकार) मध्ये वर्णन केलेली प्रतिमा रास्टर इमेजमध्ये रूपांतरित केली जाते. (पिक्सेल किंवा डॉट्स) व्हिडिओ डिस्प्लेवर आउटपुटसाठी किंवा प्रिंटर किंवा बिटमॅप फाइल फॉरमॅटमध्ये स्टोरेजसाठी.

<u>स्कॅन (conversion) रूपांतरण:</u>

• सतत चित्र किंवा ग्राफिक्स ऑब्जेक्टला स्वतंत्र पिक्सेलचा संग्रह म्हणून प्रस्तुत करण्याच्या प्रक्रियेला स्कॅन रूपांतरण म्हणतात.

१.२.४ बिटमॅप आणि वेक्टर आधारित ग्राफिक्स

- संगणक ग्राफिक्सचे रास्टर किंवा बिटमॅप ग्राफिक्स आणि वेक्टर ग्राफिक्स अशा दोन श्रेणींमध्ये वर्गीकरण केले जाऊ शकते.

१.२.५ बिटमॅप ग्राफिक्स (रास्टर ग्राफिक्स):

- पिक्सेल आधारित ग्राफिक्स आहे
- बिटमॅप प्रतिमा (ज्याला रास्टर प्रतिमा देखील म्हणतात) ग्रिडमध्ये पिक्सेल बनलेले असतात. पिक्सेल हे चित्र घटक आहेत (स्वयंक्तिक रंगाचे लहान चौरस जे बनतात तुम्ही तुमच्या स्क्रीनवर काय पाहता.) रंगाचे हे सर्व छोटे चौरस एकत्र येऊन तुम्हाला दिसत असलेल्या प्रतिमा तयार होतात.
- बिटमॅप प्रतिमा आहेत ठराव अवलंबून.
-

१.२.६ वेक्टर आधारित ग्राफिक्स(वेक्टर ग्राफिक्स):

- वेक्टर ग्राफिक्समधील प्रतिमा मुळात गणितावर आधारित प्रतिमा आहेत. बिटमॅपच्या विपरीत, वेक्टर प्रतिमा पिक्सेल नमुन्यांवर आधारित नसून त्याऐवजी असतात रेषा काढण्यासाठी गणिती सूत्रे वापरा आणि वक्र असू शकतात प्रतिमा तयार करण्यासाठी एकत्रित.
- वेक्टर आधारित प्रतिमांना गुळगुळीत कडा असतात आणि त्यामुळे वक्र तयार करण्यासाठी वापरल्या जातात आणि आकार. वेक्टर प्रतिमा संपादित केल्या आहेत ओळी हाताळणे आणि वक्र जे प्रोग्राम वापरून प्रतिमा बनवतात.

फरक यांच्यातील बिटमॅप आणि वेक्टर आधारित ग्राफिक्स -

.	**बिटमॅप आधारित ग्राफिक्स**	**वेक्टर आधारित ग्राफिक्स**
१	पिक्सेल आधारित ग्राफिक्स आहे.	गणितीय आधारित ग्राफिक्स आहे.
२	बिटमॅप प्रतिमा रिझोल्यूशन अवलंबून आहेत.	वेक्टर प्रतिमा रिझोल्यूशन स्वतंत्र आहेत.
3	वेब अनुप्रयोग साठी अनुकूल आहेत .	वेब अनुप्रयोग साठी अनुकूल नाही.
४	फेरफार अवघड आहे	फेरफार सोपे आहे.
५	बिटमॅपवरून वेक्टरमध्ये रूपांतरण करणे कठीण आहे .	वेक्टरपासून बिटमॅपमध्ये रूपांतरण करणे सोपे आहे .
६	सीमा प्रारंभ झिगझॅग दिसतो	सीमा प्रारंभ गुळगुळीत दिसतो.
७	उदा. **JPEG, GIF, BMP** इ.	उदा. फॉन्ट, लोगो चिन्हे इ.

१.३ संगणक ग्राफिक्स चा उपयोग

खालील क्षेत्रात वापरले जाते

भौगोलिक माहिती प्रणाली (**GIS).**

1. डेस्कटॉप प्रकाशन (डीटीपी)
2. ग्राफिकल वापरकर्ता इंटरफेस(**GUI)**
3. डिजिटल कला.
4. मनोरंजन.
5. प्रतिमा प्रक्रिया .
6. शिक्षण आणि प्रशिक्षण

7. सादरीकरण ग्राफिक्स, वैज्ञानिक आणि अभियांत्रिकी ग्राफिक्स
8. वैद्यकीय अनुप्रयोग
9. संवाद.
10. अनुकरण
11. Computer Aided Design/ Drafting **(CAD/CADD)**

<u>विस्तारित उपयोग</u>

ग्राफिक डिझाइन:कल्पना आणि संदेशांचे दृश्य प्रतिनिधित्व तयार करण्यासाठी चिन्हे, प्रतिमा आणि/किंवा शब्द तयार करण्यासाठी आणि एकत्रित करण्यासाठी विविध पद्धती वापरल्या जातात.

•**शिक्षण:**कॉम्प्युटर सिम्युलेशन, कॉम्प्युटर मॉडेल किंवा कॉम्प्युटेशनल मॉडेल हा एक कॉम्प्युटर प्रोग्रॅम किंवा कॉम्प्युटरचे नेटवर्क आहे, जे एका विशिष्ट सिस्टीमच्या मॉडेलचे अनुकरण करण्याचा प्रयत्न करते.संगणक सिम्युलेशन अनेक नैसर्गिकांच्या गणितीय मॉडेलिंगचा एक उपयुक्त भाग बनला आहे भौतिकशास्त्रातील प्रणाली (संगणकीय भौतिकशास्त्र), रसायनशास्त्र आणि जीवशास्त्र, अर्थशास्त्र, मानसशास्त्र आणि सामाजिक विज्ञानातील मानवी प्रणाली आणि अभियांत्रिकी नवीन तंत्रज्ञानाच्या प्रक्रियेत वापर आहे.

•**व्हिडिओ खेळ**:व्हिडिओ गेम हा एक इलेक्ट्रॉनिक गेम आहे ज्यामध्ये रास्टर डिस्प्ले डिव्हाइसवर व्हिज्युअल फीडबॅक तयार करण्यासाठी वापरकर्ता इंटरफेससह परस्परसंवाद समाविष्ट असतो.व्हिडिओ गेम खेळण्यासाठी वापरल्या जाणारया इलेक्ट्रॉनिक प्रणालींना प्लॅटफॉर्म म्हणून ओळखले जाते. हे व्यासपीठ ग्राफिक्सच्या माध्यमातून तयार होते.

•**वेब डिझाइन**:

•वेब डिझाईन हे सामान्यतः हायपरटेक्स्ट किंवा हायपरमीडियाचे सादरीकरण डिझाइन करण्याचे कौशल्य आहे जे वर्ल्ड वाइड वेबद्वारे, वेब ब्राउझरद्वारे अंतिम वापरकर्त्याला वितरित केले जाते.वेब पेजेस, वेब साइट्स, वेब ऍप्लिकेशन्स किंवा वेबसाठी मल्टीमीडिया डिझाइन करण्याच्या प्रक्रियेमध्ये ॲनिमेशन, ऑथरिंग, कम्युनिकेशन डिझाइन, कॉर्पोरेट ओळख, ग्राफिक यासारख्या अनेक विषयांचा वापर केला जाऊ शकतो. रचना, मानवी-संगणक संवाद, माहिती आर्किटेक्चर, परस्परसंवाद रचना, विपणन, छायाचित्रण, शोध इंजिन ऑप्टिमायझेशन आणि टायपोग्राफी.

•**डिजिटल कला :**डिजिटल आर्ट हे सामान्यतः डिजिटल स्वरूपात संगणकावर तयार केलेल्या कलेचा संदर्भ देते. दुसरीकडे, समकालीन कलेवर लागू केलेला एक शब्द आहे जो मोठ्या प्रमाणावर उत्पादन किंवा डिजिटल मीडियाच्या पद्धती वापरतो.डिजिटल तंत्रज्ञानाच्या प्रभावाने चित्रकला, रेखाचित्र आणि शिल्पकला यासारख्या पारंपारिक क्रियाकलापांचे रूपांतर केले आहे, तर नेट आर्ट, डिजिटल इंस्टॉलेशन आर्ट आणि आभासी वास्तविकता यासारख्या नवीन प्रकारांना कलात्मक पद्धतींची मान्यता मिळाली आहे.

•संगणक अनुकरण:कॉम्प्युटर सिम्युलेशन, कॉम्प्युटर मॉडेल किंवा कॉम्प्युटेशनल मॉडेल हा एक कॉम्प्युटर प्रोग्रॅम किंवा कॉम्प्युटरचे नेटवर्क आहे, जे एका विशिष्ट सिस्टीमच्या अमूर्त मॉडेलचे अनुकरण करण्याचा प्रयत्न करते.

संगणक मदत केली डिझाइन(CAD):• इमारती, ऑटोमोबाईल, विमान, अंतराळयान, संगणक, कापड आणि इतर अनेक उत्पादने.

•माहिती व्हिज्युअलायझेशन:माहिती व्हिज्युअलायझेशन म्हणजे सॉफ्टवेअर सिस्टीममधील फाइल्स आणि कोडच्या ओळींसारख्या मोठ्या प्रमाणातील गैर-संख्यात्मक माहितीच्या संकलनाच्या दृश्य प्रतिनिधित्वाचा अभ्यास आणि लोकांना डेटा समजण्यात आणि विश्लेषण करण्यात मदत करण्यासाठी ग्राफिकल तंत्रांचा वापर.

•प्रतिमा प्रक्रिया करणे:इमेज प्रोसेसिंग म्हणजे विद्यमान चित्रांमध्ये बदल किंवा अर्थ लावण्यासाठी तंत्रे लागू करणे. हे वैद्यकीय अनुप्रयोगांमध्ये मोठ्या प्रमाणावर वापरले जाते .

•संगणकीय भौतिकशास्त्र:कॉम्प्युटेशनल फिजिक्स हे भौतिकशास्त्रातील समस्या सोडवण्यासाठी संख्यात्मक अल्गोरिदमचा अभ्यास आणि अंमलबजावणी आहे ज्यासाठी एक परिमाणात्मक सिद्धांत आधीच अस्तित्वात आहे.हे सहसा सैद्धांतिक भौतिकशास्त्राची उपशाखा म्हणून ओळखले जाते परंतु काही लोक याला सैद्धांतिक आणि प्रायोगिक तिकशास्त्रातील मध्यवर्ती शाखा मानतात.

•मनोरंजन:संगणक ग्राफिक्स आम्हाला ॲनिमेटेड चित्रपट आणि गेम तयार करण्यास अनुमती देतात. हे दृश्ये तयार करण्यासाठी वापरले जाते विशेष प्रभाव आणि ॲनिमेशन आहेत.

<u>१.४ डिस्प्ले उपकरणे</u>:डिस्प्ले डिव्हाइस हे आउटपुट डिव्हाइस आहे जे प्रतिमाच्या (दृश्य स्वरूपात) माहितीचे प्रतिनिधित्व करण्यासाठी वापरले जाते. डिस्प्ले सिस्टमला मुख्यतः व्हिडिओ मॉनिटर किंवा व्हिडिओ डिस्प्ले युनिट (**VDU)** म्हणतात. डिस्प्ले डिव्हाइसेस मॉडेल, डिस्प्ले, पाहण्यासाठी किंवा माहिती प्रदर्शित करण्यासाठी डिझाइन केलेले आहेत. डिस्प्ले साधन आहे व्हिज्युअल हेतूसाठी प्रतिमा किंवा मजकूर यासारख्या माहितीच्या सादरीकरणासाठी वापरलेले डिव्हाइस,म्हणजे माहिती प्रदर्शित करणे.

संगणक ग्राफिक्स मध्ये परस्पर साधने:या उपकरणांमध्ये हे समाविष्ट आहे**:**

कीबोर्ड**.**,माऊस ,ट्रॅकबॉल,स्पेसबॉल,जॉयस्टिक,हलका पेन.,डिजिटायझर,टच पॅनेल**.**

संगणक ग्राफिक्स मध्ये डिस्प्ले उपकरणे

१. कॅथोड-रे ट्यूब (सीआरटी)

२.फ्लॅट पॅनल डिस्प्ले

२.१ लिक्विड क्रिस्टल डिस्प्ले (एलसीडी)

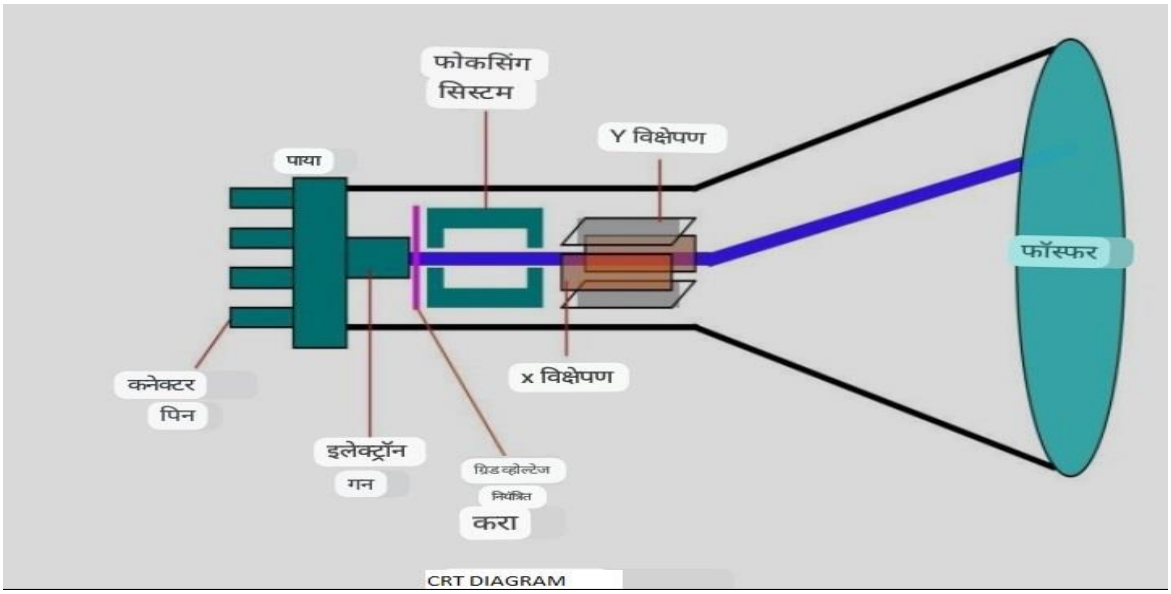
२.२. प्रकाश उत्सर्जक डायोड (एलईडी)

३. प्लाझ्मा डिस्प्ले

४. टच स्क्रीन डिस्प्ले

१.**रास्टर स्कॅन डिस्प्ले:**रास्टर स्कॅनमध्ये, इलेक्ट्रॉन गनमधून इलेक्ट्रॉन बीम फॉस्फरच्या एका ओळीत वरपासून खालपर्यंत आडवा वळवला जातो.इलेक्ट्रॉन बीम संपूर्ण स्क्रीनवर डावीकडून उजवीकडे मागे आणि पुढे

१. कॅथोड-रे ट्यूब (सीआरटी):- **CRT** म्हणजे कॅथोड रे ट्यूब. **CRT** हे पारंपारिक संगणक मॉनिटर्स आणि टेलिव्हिजनमध्ये वापरले जाणारे तंत्रज्ञान आहे. **CRT** डिस्प्लेवरील प्रतिमा स्क्रीनच्या पुढील बाजूस असलेल्या फॉस्फरसच्या नळीच्या मागील बाजूस इलेक्ट्रॉन्स फायर करून तयार केली जाते.एकदा इलेक्ट्रॉनने फॉस्फरस गरम केल्यावर ते उजळतात आणि ते स्क्रीनवर प्रक्षेपित होतात. तुम्ही स्क्रीनवर पाहत असलेला रंग लाल, निळा आणि हिरव्या प्रकाशाच्या मिश्रणाने तयार होतो. इलेक्ट्रॉन बंदूक उत्सर्जित करते एक तुळई इलेक्ट्रॉनचे (कॅथोड किरण).इलेक्ट्रॉन बीम फोकसिंग आणि डिफ्लेक्शन सिस्टममधून जातो जे त्यास निर्दिष्ट दिशेने निर्देशित करतात फॉस्फर-लेपित स्क्रीनवरील पोझिशन्स.जेव्हा बीम स्क्रीनवर आदळतो, तेव्हा फॉस्फर संपर्क केलेल्या प्रत्येक स्थानावर प्रकाशाचा एक छोटासा स्पॉट उत्सर्जित इलेक्ट्रॉन बीम द्वारे करतो.



आकृती१_सी आर टी मॉनिटर

<u>सी आर टी मॉनिटर टाईप</u> :

१:मोनोक्रोम सी आर टी

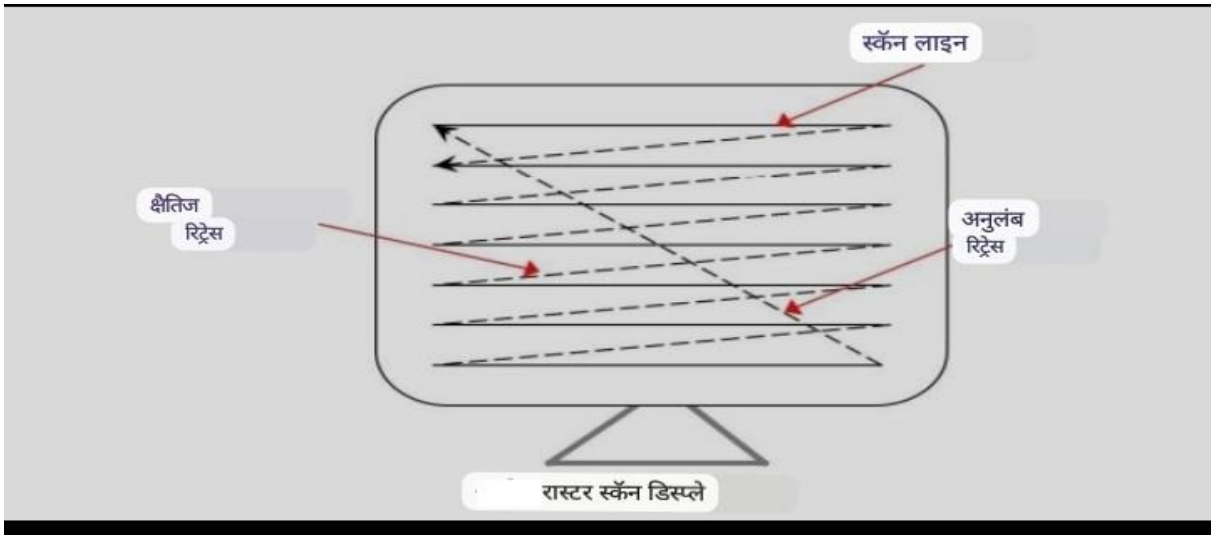
२.कलर सी आर टी

<u>कॅथोड रे ट्यूब तेथे दोन तंत्र वापरले आहेत :-</u>

रास्टर स्कॅन डिस्प्ले

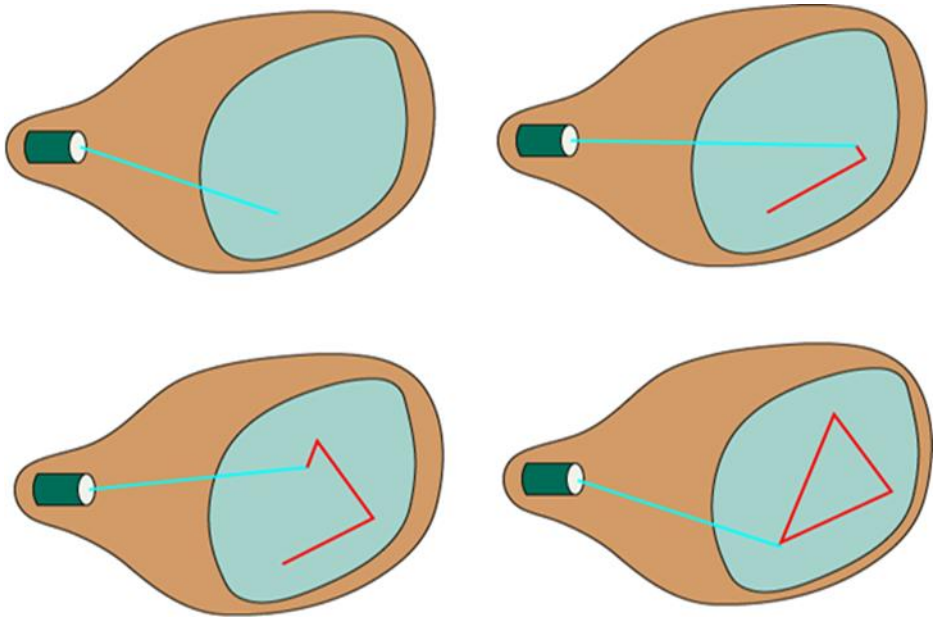
वेक्टर स्कॅन डिस्प्ले

<u>रास्टर स्कॅन डिस्प्ले</u> त्याच प्रतिमेचे स्कॅनिंग पुनरावृत्ती करून स्क्रीनवरील स्थिर प्रतिमा राखतात. ही प्रक्रिया स्क्रीन रिफ्रेशिंग म्हणून ओळखली जाते.



आकृती२ रास्टर स्कॅन डिस्प्ले

2.**वेक्टर स्कॅन डिस्प्ले:**या तंत्रात, इलेक्ट्रॉन बीम रास्टर स्कॅनप्रमाणे डावीकडून उजवीकडे आणि वरपासून खालपर्यंत स्कॅन करण्याऐवजी स्क्रीनच्या फक्त त्या भागाकडे निर्देशित केला जातो जिथे चित्र काढायचे आहे.ते आहे तसेच म्हणतात वेक्टर प्रदर्शन, स्ट्रोक-लेखन प्रदर्शन, किंवा कॅलिग्राफिक प्रदर्शन सेकंदाला 30 ते 60 वेळा चित्राच्या सर्व घटक रेषा काढण्यासाठी डिझाइन केलेले आहेत . रँडम-स्कॅन डिस्प्लेमध्ये इलेक्ट्रॉन बीम फक्त स्क्रीनच्या त्या भागात निर्देशित केला जातो जिथे चित्र काढायचे असते. याला वेक्टर डिस्प्ले असेही म्हणतात, कारण ते एका वेळी एक रेषा काढते. हे कोणत्याही निर्दिष्ट अनुक्रमात चित्राच्या घटक रेषा काढू आणि रीफ्रेश करू शकते. पेन प्लॉटर हे यादृच्छिक-स्कॅन डिस्प्लेचे उदाहरण आहे.



आकृती ३ रँडम-स्कॅन डिस्प्ले

फरक करा यांच्यातील रँडम स्कॅन आणि रास्टर स्कॅन.

1. रिझोल्यूशन :रँडम स्कॅनचे रिझोल्यूशन रास्टर स्कॅनपेक्षा जास्त आहे. रास्टर स्कॅनचे रिझोल्यूशन :रँडम स्कॅनपेक्षा कमी किंवा कमी असते.

2. खर्च :हे रास्टर स्कॅनपेक्षा महाग आहे. रास्टर स्कॅनची किंमत :रँडम स्कॅनपेक्षा कमी आहे.

3. बदल:रँडम स्कॅनमध्ये, रास्टर स्कॅनच्या तुलनेत कोणताही बदल करणे सोपे आहे. रास्टर स्कॅनमध्ये असताना, कोणताही बदल करणे इतके सोपे नसते.

4. इंटरलेसिंग: :रँडम स्कॅनमध्ये, इंटरलेसिंग वापरले जात नाही. रास्टर स्कॅनमध्ये असताना, इंटरलेसिंग वापरले जाते.

5. रेखाचित्रे:रँडम स्कॅनमध्ये, प्रतिमा किंवा चित्र रेंडरिंगसाठी गणितीय कार्य वापरले जाते. हे बहुभुज रेखाचित्रे आवश्यक असलेल्या अनुप्रयोगांसाठी योग्य आहे. ज्यामध्ये, प्रतिमा किंवा चित्र रेंडरिंगसाठी, रास्टर स्कॅन पिक्सेल वापरतो. हे वास्तववादी दृश्ये तयार करण्यासाठी योग्य आहे.

6. इलेक्ट्रॉन बीमची हालचाल :इलेक्ट्रॉन बीम स्क्रीनच्या फक्त त्या भागाकडे निर्देशित केली जाते जिथे चित्र काढणे आवश्यक आहे, एका वेळी एक रेषा. इलेक्ट्रॉन बीम वरपासून खालपर्यंत आणि स्क्रीनवर एका वेळी एक पंक्ती निर्देशित केली जाते. हे संपूर्ण स्क्रीनवर निर्देशित केले जाते.

7. पिक्चर डेफिनिशन: हे रिफ्रेश बफरमध्ये पिक्चर डेफिनिशनला लाइन कमांड्सचा संच म्हणून संग्रहित करते. हे फ्रेम बफरमध्ये पिक्सेलच्या तीव्रतेच्या मूल्यांचा संच म्हणून चित्र व्याख्या संग्रहित करते.

8. रिफ्रेश रेट :रिफ्रेश रेट प्रदर्शित करण्याच्या ओळींच्या संख्येवर अवलंबून असतो, म्हणजेच प्रति सेकंद 30 ते 60 वेळा. रिफ्रेश दर 60 ते 80 फ्रेम्स प्रति सेकंद आहे आणि चित्र जटिलतेपासून स्वतंत्र आहे.

9. सॉलिड पॅटर्न ::रँडम स्कॅनमध्ये, सॉलिड पॅटर्न भरणे कठीण असते. रास्टर स्कॅनमध्ये, सॉलिड पॅटर्न भरणे सोपे आहे.

10. उदाहरण :रँडम स्कॅन-पेन प्लॉटर, रास्टर स्कॅन-टीव्ही सेट.

<u>2.फ्लॅट पॅनल डिस्प्ले</u>

2.1 लिक्विड क्रिस्टल डिस्प्ले (एलसीडी): नॉन-इमिसिव्ह डिस्प्ले प्रकाशीय प्रभावांचा वापर करून सूर्यप्रकाश किंवा इतर स्रोतातील प्रकाशाचे ग्राफिक्स पॅटर्नमध्ये रूपांतर करतात. एलसीडी (लिक्विड क्रिस्टल डिव्हाइस) ही उदाहरणे आहेत.

<u>एलसीडी डिस्प्ले:</u>

•लिक्विड क्रिस्टल डिस्प्ले ही अशी उपकरणे आहेत जी सभोवतालच्या ध्रुवीकृत प्रकाशातून किंवा अंतर्गत प्रकाश स्रोतातून द्रव-क्रिस्टल सामग्रीद्वारे एक चित्र तयार करतात.

•एलसीडी दोन ग्लास प्लेट्समध्ये द्रव-क्रिस्टल सामग्री वापरते; प्रत्येक प्लेट द्रव भरलेल्या प्लेट्स दरम्यान

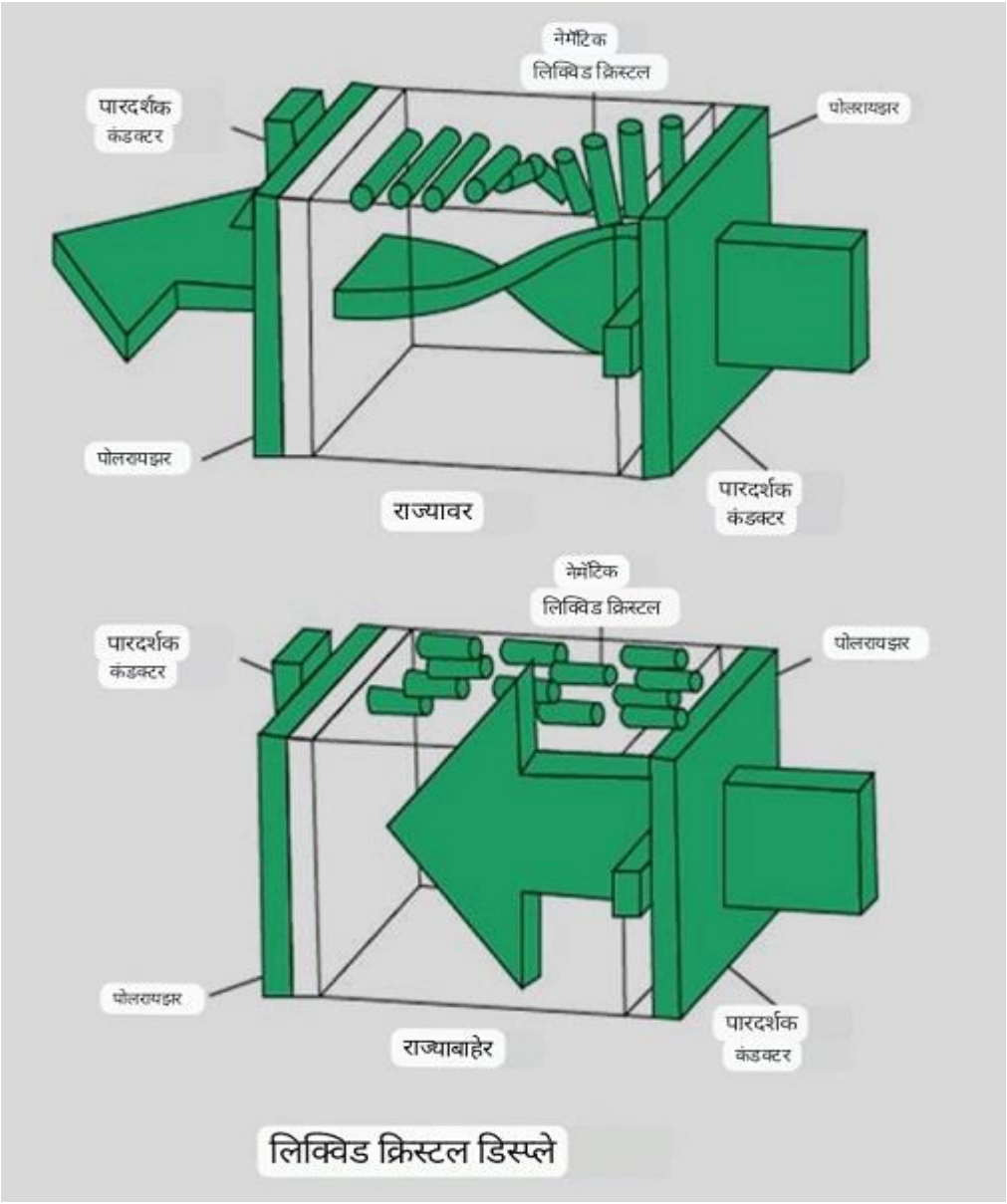
एकमेकांना काटकोन आहे.

•एक काच प्लेट कंडक्टर व्यवस्था समावेश पंक्ती च्या कंडक्टर व्यवस्था केली जाते.

•दुसरा काच प्लेट मध्ये पंक्ती चि क्षैतिज दिशा ठरवली जाते.

•पिक्सेलची स्थिती उभ्या आणि क्षैतिज कंडक्टरच्या छेदनबिंदूद्वारे निर्धारित केली जाते. ही स्थिती स्क्रीनचा सक्रिय भाग आहे.

•लिक्विड क्रिस्टल डिस्प्ले तापमानावर अवलंबून असते. ते शून्य ते सत्तर अंश सेल्सिअस दरम्यान आहे. ते सपाट आहे आणि ऑपरेट करण्यासाठी खूप कमी शक्ती लागते.



आकृती ४_एलसीडी डिस्प्ले

2.2. प्रकाश उत्सर्जक डायोड (एलईडी): उत्सर्जित डिस्प्ले ही अशी उपकरणे आहेत जी विद्युत उर्जेचे प्रकाशात रूपांतर करतात. प्लाझ्मा पॅनेल**,** पातळ फिल्म इलेक्ट्रोल्युमिनेसेंट डिस्प्ले आणि **LED (**लाइट इमिटिंग डायोड्स) ही उदाहरणे आहेत.

एलईडी डिस्प्ले:

•_एलईडी डिस्प्ले मध्ये एक एलईडी, मॅट्रिक्स च्या डायोड आहे आयोजित करण्यासाठी फॉर्म द पिक्सेल पोझिशन्स मध्ये द प्रदर्शन आणि चित्र व्याख्या रिफ्रेश बफरमध्ये संग्रहित केली जाते.

•रिफ्रेश बफरमधून डेटा वाचला जातो आणि डिस्प्लेमध्ये प्रकाश नमुना तयार करण्यासाठी डायोडवर लागू केलेल्या व्होल्टेज स्तरांमध्ये रूपांतरित केले जाते.

•एलसीडी हलके, संक्षिप्त, पोर्टेबल स्वस्त, अधिक विश्वासार्ह, आणि वापरायलासोपे आहेत.

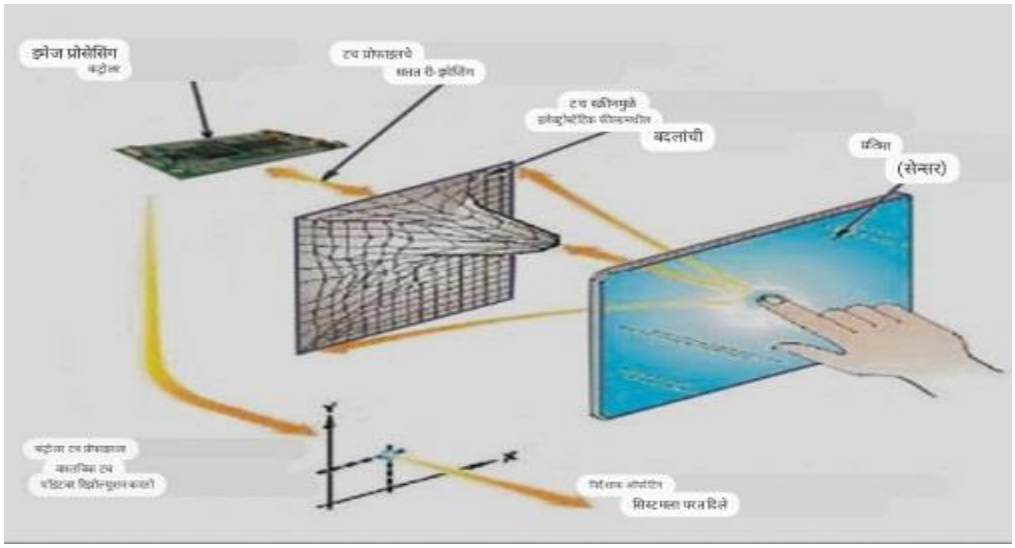
•एलसीडी पातळ थर च्या द्रव क्रिस्टल, द्रव ते प्रदर्शन क्रिस्टलीय गुणधर्म वापर करतो.

• एलईडी डिस्प्ले मध्ये दोन इलेक्ट्रिकली आयोजित प्लेट्स सँडविच केलेले आहेत.

•शीर्ष प्लेट पारदर्शक इलेक्ट्रोड जमा करते, आणि परत प्लेट एक परावर्तक किंवा मागून प्रकाशित प्रकाश प्रदर्शित करते जेणेकरून दर्शक स्क्रीनवरील प्रतिमा पाहू शकतील.

•सर्व प्लेट्सवर नियंत्रित विद्युत सिग्नल लागू करून, लिक्विड क्रिस्टलचे विविध विभाग सक्रिय केले जाऊ शकतात, ज्यामुळे त्यांच्या प्रकाशाचा प्रसार किंवा ध्रुवीकरण गुणधर्मांमध्ये बदल होतात.

• विविध इलेक्ट्रॉनिक्स सारखे घड्याळे, कॅल्क्युलेटर आणि नोटबुक संगणक मध्ये वापरले आहेत



आकृती ५_एलईडी डिस्प्ले

3. प्लाझ्मा पॅनल डिस्प्ले:

•प्लाझ्मा-पॅनल्स ला गॅस-डिस्चार्ज डिस्प्ले असे पण म्हणतात.याचा उपयोग दिवे फ्लोरोसंट मध्ये होतो. प्लाझ्मा-पॅनेल डिस्प्लेचे आवश्यक घटक आहेत:

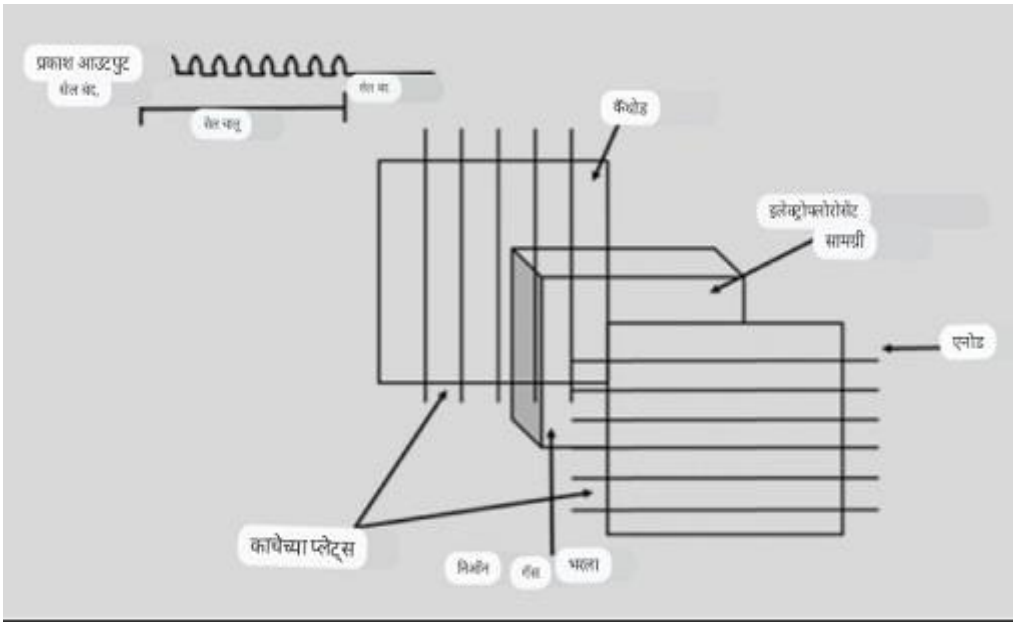
•कॅथोड: त्यात बारीक तारांचा समावेश होतो. हे गॅस पेशींना नकारात्मक व्होल्टेज वितरीत करते. व्होल्टेज सोडले जाते नकारात्मक अक्षासह.

•एनोड: ते तसेच समावेश आहे च्या ओळ तारा ते वितरित करते सकारात्मक विद्युतदाब. विद्युतदाब आहे पुरवले सकारात्मक अक्षासह.

•फ्लोरोसेंट पेशी: गॅस द्रव ,विद्युतदाब लागू करण्यासाठी हा द्रव (निऑन गॅस) तो प्रकाश उत्सर्जित करतो.

•जेव्हा क्षैतिज आणि उभ्या तारांमध्ये लक्षणीय व्होल्टेज फरक असेल तेव्हा गॅस मंद होईल. व्होल्टेज पातळी 90 व्होल्ट ते 120 व्होल्ट दरम्यान ठेवली जाते. प्लाझ्मा पातळीला रीफ्रेश करण्याची आवश्यकता नाही .

•प्लाझ्मा पॅनेलमधील प्रदर्शित करण्यायोग्य बिंदू क्षैतिज आणि अनुलंब ग्रिडच्या क्रॉसिंगद्वारे बनविला जातो. प्लाझ्मा पॅनेलचे रिझोल्यूशन 512 * 512 पिक्सेल पर्यंत असू शकते.



आकृती ६_प्लाझ्मा पॅनल डिस्प्ले

4:स्पर्श (Touch) स्क्रीन:

•टच स्क्रीन एक संगणक डिस्प्ले स्क्रीन आहे जी इनपुट डिव्हाइस म्हणून कार्य करते. जेव्हा टच स्क्रीन असते बोटाने किंवा लेखणीने स्पर्श केल्यावर, ते इव्हेंटची नोंदणी करते आणि प्रक्रियेसाठी कंट्रोलरकडे पाठवते.

• स्पर्श स्क्रीन आहे दोन मुख्य फायदे

•प्रथम, हे वापरकर्त्यांना माउस किंवा टचपॅडद्वारे नियंत्रित पॉइंटरसह अप्रत्यक्षपणे न दाखवता जे प्रदर्शित केले जाते त्याच्याशी थेट संवाद साधण्याची परवानगी देते.

•याचा वापर खेळणे प्रमुख भूमिका मध्ये डिझाइन च्या डिजिटल साधने अशा म्हणून वैयक्तिक डिजिटल असिस्टंट (पीडीए), सॅटेलाइट नेव्हिगेशन उपकरणे, मोबाईल फोन आणि व्हिडिओ गेम्स मध्ये केला जातो.

• तीन मुख्य स्पर्श स्क्रीन तंत्रज्ञान आहेत:

•रेझिस्टिव्ह : या स्क्रीनमध्ये पातळ धातूचा थर असतो जो प्रवाहकीय आणि प्रतिरोधक असतो, त्यामुळे परिणाम स्पर्श होतो.

•पृष्ठभाग ध्वनिक लहरी (**SAW):** प्रचंड कंपनसंख्या असलेल्या (ध्वनिलहरी) लाटा या स्क्रीनवरून जातात. त्याला स्पर्श केल्याने लहरीचा काही भाग शोषून घेतला जातो, स्पर्शाची स्थिती नोंदणीकृत होते, जी कंट्रोलरला पाठविली जाते. साधक: बोट किंवा लेखणीला प्रतिसाद देते. बाधक: धूळ किंवा पाण्यामुळे नुकसान होऊ शकते.

•कॅपेसिटिव्ह: ही स्क्रीन इलेक्ट्रिकली चार्ज केलेल्या सामग्रीने लेपित आहे. त्याला स्पर्श केल्याने कॅपेसिटन्समध्ये बदल होतो, ज्यामुळे स्थान निश्चित केले जाऊ शकते आणि कंट्रोलरला पाठवले जाऊ शकते. साधक: धूळ किंवा पाण्याने नुकसान होत नाही आणि उच्च स्पष्टता आहे. बाधक: फक्त बोटाने स्पर्श करणे आवश्यक आहे - एक लेखणी वापरली जाऊ शकत नाही.

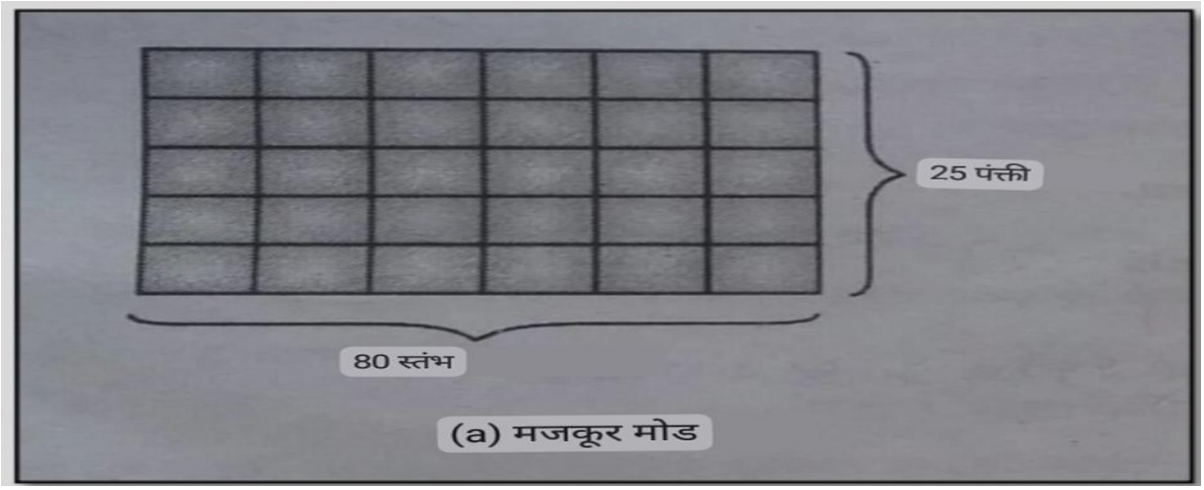
आकृती ७_स्पर्श (Touch**) स्क्रीन**

1.5टेक्स्ट मोड आणि ग्राफिक्स मोड

- मुळात आहेत दोन प्रकारचे ग्राफिक्स मोड म्हणजे, **टेक्स्ट मोड** आणि **ग्राफिक्स मोड.**
- तेथे आहेत वेगळे ग्राफिक्स कार्ये उपलब्ध मध्ये या दोन मोड उपयुक्त च्या साठी विविध प्रभावी मजकूर आणि विविध प्रकारचे भौमितिक आकार रेखाटणे.

1.टेक्स्ट मोड

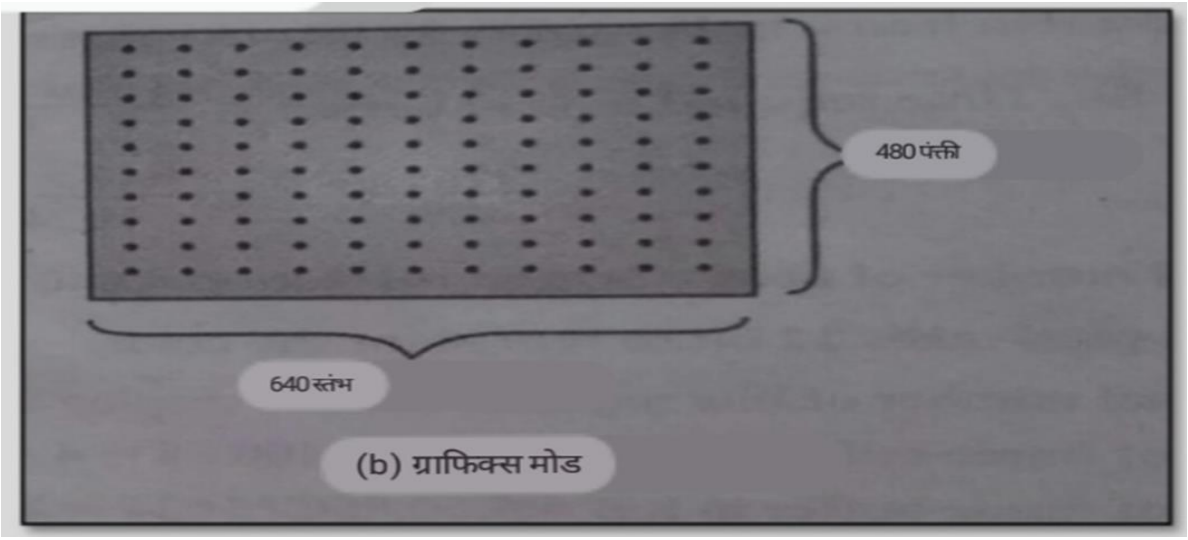
- टेक्स्ट मोडमध्ये, डिस्प्ले स्क्रीन बॉक्सच्या पंक्ती आणि स्तंभांमध्ये विभागली जाते. प्रत्येकामध्ये एक वर्ण असू शकतो. टेक्स्ट मोडला अक्षर मोड देखील म्हणतात.
- सर्व व्हिडिओ मजकूर मोडला समर्थन देतात जे स्क्रीनला 25 **rows** पंक्ती आणि 80 **column** स्तंभांमध्ये विभाजित करते.



आकृती ८ टेक्स्ट मोड

2. ग्राफिक्स मोड

- ग्राफिक्स मोडमध्ये, डिस्प्ले स्क्रीनचा अॅरे मानला जातो पिक्सेल कार्यक्रम जे ग्राफिक्स मोडमध्ये रन केल्याने अमर्यादित विविध आकार आणि फॉन्ट मिळतात.



आकृती ९ ग्राफिक्स मोड

टेक्स्ट मोड ग्राफिक्स चे FUNCTION

1) Window( ):- This function specifies a window on screen. The four integer coordinates of the window are passed as parameters to this function.

(Syntax) कस लिहायचं आहे - window (left, top, right, bottom);

2) putch( ): It displays a single character at cursor position.

Syntax – putch(char);

Example- putch('A');

Displays character A at specified cursor position.

3) clrscr( ): It clears the entire screen and locates the cursor in top left corner of screen i.e(1,1).

Syntax – clrscr( );

4) gotoxy( ): It positions the cursor to the specified location on screen, where location is specified by x, y co-ordinates of the point.

Syntax – gotoxy(x, y);

Where x, y are co-ordinates of a point where cursor is to be positioned.

Example: gotoxy(3, 4);

It positions cursor to 3rd row, 4th column.

5) puts( ): It display string at cursor position.

Syntax – puts(s1);

Where, s1 is string to be displayed at specified cursor position. For puts( ) and putch( ) functions, gotoxy( ) can be used.

Example: gotoxy(3, 4);

puts("Hello");

gotoxy(10,10);

putch('A')

Here, Hello is displayed starting from the position 3,4 and 'A' is displayed at 10,10.

6) textcolor( ): It sets the colour for the text. Any text displayed after this command will be displayed in a colour specified by this command. The supported colours are

रंग स्थिर	रंग नाव
0	काळा
१	निळा
2	हिरवा
3	निळसर

4	लाल
५	मॅजेन्टा
6	तपकिरी
७	प्रकाश राखाडी
8	गडद राखाडी
९	प्रकाश निळा
10	प्रकाश हिरवा
11	प्रकाश निळसर
12	प्रकाश लाल
13	प्रकाश मॅग्नेटा
14	पिवळा
१५	पांढरा
128	**BLINK**

Table no 1:टेबल१

Syntax: textcolor (color);

Example: textcolor('GREEN');

Is equivalent to int col= 2;

textcolor(col);

Above both codes displays subsequent texts in green colour.

7) deline( ): It delets a line specified by cursor position. After deletion, all subsequent lines will be pushed up by one line.

Syntax: deline( );

Example: gotoxy(12, 4);

delline( );

Above statement delets 8th line.

8) inline( ): It inserts a blank line at current cursor position.

Syntax: inline( );

Example : gotoxy(8, 4);

inline( );

It inserts a line at 8th row.

9) textbackgroud( ):It changes background colour of text. The valid colour for

CGA(Color Graphics Adapter) are from 0 to 6. They are:

Constant Colour Name

0 BLACK

1 BLUE

2 GREEN

3 CYAN

4 RED

5 MAGENTA

6 BROWN

Syntax: textbackground(color);

Example : int col = 4

textbackground(col);

It sets text's background colour to red

10) moveto( ): It moves cursor to the location specified by int(x, y) co- ordinates.

Syntax: moveto(x, y);

11) outtextxy( ): ("sentence")

OR

Outtextxy(x, y, "sentence")

Where, x, y gives co-ordinates of a point where from to display text.

It displays text within quotation mark at specified position and with latest setcolour

style.

ग्राफिक्स मोड ग्राफिक्स चे (Graphics Mode)Functions:

1)initgraph() Function: It is used to set the graphics mode. It has the following form:

initgraph(graphic_driver,graphic_mode,driver_path);

2)closegraph() Function: It is used to terminate the graphics mode which is set by the initgraph() function. It restores the original video mode. It has the following form:

closegraph();

3)getmaxx() Function: This function is used to get the maximum value of x coordinate in the current resolution.

Example:

max=getmaxx();

setcolor() Function: It is used to specify the colour for a graphical picture going to be displayed. It has the following form:

4)setcolor(m); // where m is an integer or symbol representing the colour of the drawing.

5)setbkcolor() Function: This function is used to set the background color for the monitor screen. It has the following form:

setbkcolor(m);

6)putpixel() Function: This function is used to draw a point on the monitor screen. It has the following form:

putpixel(x,y,n);

7)line() Function: It is used to draw a line on the monitor screen in the given coordinate position. It has the following form:

line(x1, y1, x2, y2);

8)outtextxy() Function: This function is used to display a text message in a specific location on the monitor screen. It has the following form:

outtextxy(m, n, "text message"); // where m specifies the position in x-axis and n specifies

the position in y-axis.

9)rectangle() Function: It is used to draw a rectangle by specifying the diagonal coordinates. It

has the following syntax:

rectangle(x1, y1, x2, y2); // where x1 and y1 are integers representing the top-left coordinate of the rectangle. x2 and y2 are integers representing the bottom-right coordinates of the rectangle.

10)circle() Function: It is used to draw a circle by specifying its center and radius. It has the following syntax:

circle(x, y, r)

11)arc() Function: This function is used to draw a circular arc by specifying the centre, radius and starting and ending angles. It has the following syntax:

arc(x, y, as, ea, r)

12)ellipse() Function: This function is used to draw an ellipse by specifying the centre, radius, start and end angles and the semi-major and semi-minor axes of the ellipse. It has the following syntax:

ellipse(x, y, as, ea, xr, yr);

13)bar() Function: It is used to draw a rectangular bar using the diagonally opposite corners and fill it with current fill style and colour. It has the following syntax:

bar(x1, y1, x2, y2);

14)cleardevice() Function: It is used to clear the screen in graphics mode. It has the following syntax:

cleardevice();

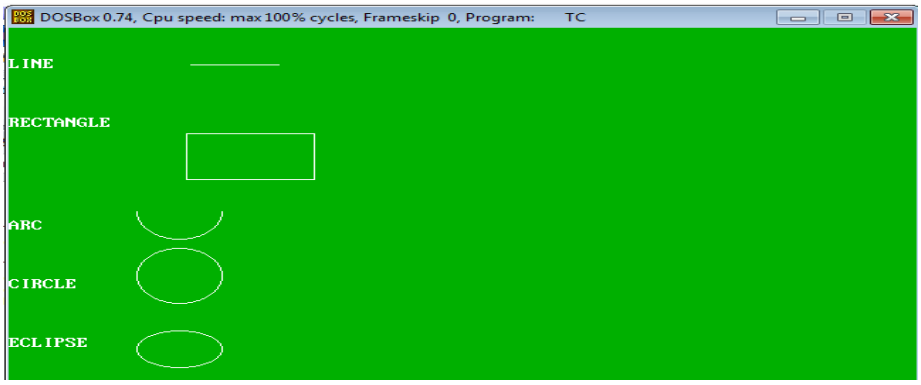
उदाहरण सॅम्पल प्रोग्रॅम

ग्राफिक्स चे बेसिक शेप काढणे

1:Write a Program to draw basic graphics construction like line, circle, arc, ellipse and rectangle.

```
#include<graphics.h>
#include<conio.h>
void main()
{
   intgd=DETECT,gm;
```

```
    initgraph (&gd,&gm,"c:\\tc\\bgi");
    setbkcolor(GREEN);
    printf("\t\t\t\n\nLINE");
    line(50,40,190,40);
    printf("\t\t\n\n\n\nRECTANGLE");
        rectangle(125,115,215,165);
    printf("\t\t\t\n\n\n\n\n\n\nARC");
    arc(120,200,180,0,30);
    printf("\t\n\n\n\nCIRCLE");
    circle(120,270,30);
    printf("\t\n\n\n\nECLIPSE");
    ellipse(120,350,0,360,30,20);
    getch();
}
```



2. C program to draw circle built-in graphics commands.

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
int main( )
```

```
{
int gd= DETECT, gm;
int x, y, radius=80;
initgraph(&gd, &gm, "C:\\Turboc3\BGI");
/* Initialize center of circle with center of screen */
x = getmaxx( ) / 2;
y= getmaxy( ) / 2;
outtextxy(x-100, 50, "Circle using Graphics in C");
/*Draw circle on screen*/
circle(x, y, radius);
getch( );
closegraph( );
return 0;
};
```

3. Simple program to draw circle.

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
int main( )
{
int gd= DETECT, gm;
initgraph(&gd, &gm, "C:\\Turboc3\BGI");
circle(25, 25, 100);
getch( );
closegraph( );
return 0;
}
```

4. C program to draw rectangle and a line.

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
int main( )
{
int gd= DETECT, gm;
initgraph(&gd, &gm, "C:\\Turboc3\BGI");
/* Draw rectangle on screen */
rectangle(150, 50, 400, 150);
/* Draw line on screen */
Line(500, 150, 600, 250);
getch( );
closegraph( );
return 0;
}
```

5. Program to draw polygon.

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
int main( )
{

int gd= DETECT, gm;
int points[ ]= {320, 150, 420, 300, 250, 300, 320, 150 };
initgraph(&gd, &gm, "C:\\Turboc3\BGI");
```

```
drawpoly(4, points);
getch( );
closegraph( );
return 0;
}
```

6. C program to draw a circle and fill it by color and pattern.

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
int main( )
{
int gd= DETECT, gm;
initgraph(&gd, &gm, "C:\\Turboc3\BGI");
circle(100, 100, 50);
setfillstyle(HATCH_FILL, RED); // set fill pattern and color
floodfill(100, 100, RED); // 100, 100 is just a point inside circle
getch( );
closegraph( );
}
```

१.६ संगणक ग्राफिक्समध्ये फाइल आणि त्याची रचना (Display File Structure)

डिस्प्ले फाइल ही ग्राफिक्स कमांडची मालिका आहे जी आउटपुट इमेज परिभाषित करते. विविध प्रिमिटिव्स एकत्र करण्यासाठी कमांड्स कार्यान्वित करून प्रतिमा तयार केली जाते .

i:डिस्प्ले प्रोसेसर

डिस्प्ले प्रोसेसर हा संगणक ग्राफिक्सचा एक भाग आहे जो कोडला चित्रांमध्ये रूपांतरित करण्यासाठी वापरला जातो. दुसऱ्या शब्दांत, आपण असे म्हणू शकतो की डिस्प्ले प्रोसेसरचा वापर डिजिटल माहिती किंवा डिजिटल सिग्नलला ॲनालॉगमध्ये रूपांतरित करण्यासाठी केला जातो. म्हणून, डिस्प्ले प्रोसेसरला सोप्या भाषेत डिजिटल ते

ॲनालॉग कनवर्टर देखील म्हटले जाऊ शकते. डिस्प्ले प्रोसेसरमधील रूपांतरण उपकरणांच्या प्रकारांवर आणि ग्राफिकल प्रस्तुतीकरणासाठी वापरल्या जाणाऱ्या कार्यांवर अवलंबून असते.डिस्प्ले प्रोसेसरचा मुख्य उद्देश स्कॅन रूपांतरण आहे. स्कॅन रूपांतरण ही पिक्सेलचा संग्रह म्हणून भिन्न ग्राफिक्स ऑब्जेक्ट्स प्रदर्शित करण्याची प्रक्रिया आहे. या प्रक्रियेत, आपल्याला लंबवर्तुळ, आयत आणि बहुभुज यांसारख्या वेगवेगळ्या गणितीय आकारांचा संग्रह असलेल्या ग्राफिक्स ऑब्जेक्ट्समध्ये फरक करावा लागेल. त्याला डिस्प्ले प्रोसेसिंग युनिट, थोडक्यात डीपीयू असेही म्हणतात.

Ii:डिस्प्ले प्रोसेसरचे भाग

डिस्प्ले प्रोसेसरचे **4** प्रमुख भाग आहेत,

A:फाइल मेमरी प्रदर्शित करा

B:डिस्प्ले कंट्रोलर

C:डिस्प्ले जनरेटर

D:डिस्प्ले कन्सोल

आता आपण या प्रत्येक भागाचा वापर आणि कार्य याबद्दल अभ्यास करू.

A) फाइल मेमरी प्रदर्शित करा

याचा वापर स्क्रीनवर चित्रे दाखवण्यासाठी केला जातो. डिस्प्ले फाइल मेमरी विविध ग्राफिक्स वस्तू किंवा संस्था ओळखते. स्क्रीनवर दर्शविल्या जाणाऱ्या सर्व पिक्सेल मूल्ये डिस्प्ले फाइल मेमरीमध्ये आहेत.

B) डिस्प्ले कंट्रोलर

डिस्प्ले कंट्रोलर डिस्प्ले प्रोसेसरमधील प्रवाह सूचना हाताळतो. हे व्यत्यय हाताळण्यासाठी आणि त्यांच्या अंमलबजावणी दरम्यानची वेळ राखण्यासाठी जबाबदार आहे. सूचनांचे स्पष्टीकरण डिस्प्ले कंट्रोलरद्वारे देखील केले जाते.

C) डिस्प्ले जनरेटर

डिस्प्ले जनरेटरचा वापर स्क्रीनवर वर्ण आणि आकार निर्माण करण्यासाठी केला जातो. हे आम्ही दिलेल्या इनपुटनुसार आकार आणि वर्ण प्रदर्शित करते. थोडक्यात, स्क्रीनवर वक्र आणि अक्षरे निर्माण करण्यासाठी डिस्प्ले जनरेटरचा वापर केला जातो.

D) डिस्प्ले कन्सोल

डिस्प्ले कन्सोल हे डिस्प्ले डिव्हाइस आणि इनपुट डिव्हाइसचे संयोजन आहे. हा डिस्प्ले प्रोसेसरचा भाग आहे जो स्क्रीनवर आउटपुट दाखवतो. कॅथोड रे ट्यूब डिस्प्ले कन्सोल अंतर्गत येते. सामान्य शब्दात, डिस्प्ले कन्सोल ही स्क्रीन आहे ज्यावर तुम्ही तुमचे सर्व ग्राफिकल आउटपुट पाहता.

iii:प्रिमिटिव्ह ऑपरेशन्स परफॉर्म ऑन डिस्प्ले फाईल:

1:move to-move(x1,y1) move pont from x1 to y1.

2:lineto-line(x2,y2)(line from x2 to y2).

3:Draw-to dwaw line.

Iv:कॉम्प्युटर ग्राफिक्समध्ये डिस्प्ले फाइल इंटरप्रिटर म्हणजे काय?

1. डिस्प्ले फाइल इंटरप्रिटर हा डिस्प्ले फाइल फॉरमॅटसाठी इंटरप्रिटर आहे**,** जो संगणक ग्राफिक्समध्ये वापरला जाणारा फाइल फॉरमॅट आहे**.**

2. हे प्रोग्राम्सला प्रोग्राम म्हणून स्थापित न करता संगणकावर चालविण्यास अनुमती देते.डिस्प्ले लिस्ट **(**किंवा डिस्प्ले फाइल**)** ही ग्राफिक्स कमांडची मालिका आहे जी आउटपुट इमेज परिभाषित करते**.** प्रदर्शन सूची द्विमितीय आणि त्रिमितीय दृश्ये दर्शवू शकते**.** दृश्य संचयित करण्यासाठी प्रदर्शन सूचीचा वापर करणाऱ्या प्रणालींना तात्काळ मोड सिस्टमच्या विरूद्ध रिटेन्ड मोड सिस्टम म्हणतात**.**

१.७ **संगणक ग्राफिक्स मध्ये 2d आणि 3d समन्वय प्रणाली(Coordinate System)**

2D मध्ये दोन समन्वय वापरले जातात**,** म्हणजे **x** आणि **y** तर **3D** मध्ये **x, y** आणि **z** हे तीन समन्वय वापरले जातात**.** त्रिमितीय प्रतिमा आणि वस्तूंसाठी**,** त्रिमितीय परिवर्तन आवश्यक आहे**.** हे भाषांतर**,** स्केलिंग आणि रोटेशन आहेत**.** मॅट्रिक्स वापरून मुलभूत परिवर्तने दर्शविली जातात म्हणून याला असेही म्हणतात**.**

१.८ Standards for Computer Graphics (संगणक ग्राफिक्ससाठी standard)

1. CORE (Core of a Graphics System)

2. GKS (Graphics kernel system)

3. IGES(Initial Graphics Exchange Specifications)

4. PHIGS(Programmer’s Hierarchical Interactive Graphics System)

5. CGM(Computer Graphics Metafile)

6.CGI(ComputerGraphics Interface)

१.९ ग्राफिक्स फाईल फॉर्मँट

1. JPEG (or JPG) - Joint Photographic Experts Group
2. PNG - Portable Network Graphics

3. GIF - Graphics Interchange Format
4. TIFF - Tagged Image File
5. PSD - Photoshop Document
6. PDF - Portable Document Format
7. EPS - Encapsulated Postscript
8. AI - Adobe Illustrator Document
9. INDD - Adobe Indesign Document
10. RAW - Raw Image Formats

१.१० संगणक ग्राफिक्समधील नवीनतम ट्रेंड

१.व्हर्च्युअल रिॲलिटी (VR) : व्हर्च्युअल रिॲलिटी म्हणजे आपले संगणक जे प्रत्यक्षात अस्तित्वात नसले तरीही गोष्टींचा अनुभव घेणे. व्हर्च्युअल रिॲलिटी (**VR)** अक्षरशः काहीही अनुभवणे शक्य करते**,** कुठेही**,** कधीही. ते आहे द सर्वाधिक इमर्सिव्ह प्रकार वास्तव तंत्रज्ञान आणि मानवी मेंदूला हे पटवून देऊ शकते की ते कुठेतरी आहे ते खरोखर नाही.

हा विशेष प्रकारचा ग्राफिकल यूजर इंटरफेस आहे जो संगणकाद्वारे व्युत्पन्न केलेले इमर्सिव्ह**,** त्रिमितीय**,** परस्परसंवादी वातावरण सादर करतो ज्यात प्रवेश केला जातो आणि वापरून हाताळले जाते**,** उदाहरणार्थ**,** स्टिरिओ हेडफोन**,** हेड-माउंट केलेले स्टिरिओ टेलिव्हिजन गॉगल्स आणि डेटा-ग्लोव्ह्ज**. .**

उपकरणे वापरले मध्ये आभासी वास्तव प्रणाली.(Virtual Reality)

कन्सोल / स्मार्टफोन/संगणक,इनपुट उपकरणे,जॉयस्टिक्स, ट्रॅकिंग गोळे,ट्रॅकपॅडचालू- साधन नियंत्रण बटणे,गती ट्रॅकर्स/ बॉडीसूट,गती प्लॅटफॉर्म

ॲप्लिकेशन

शिक्षण

मनोरंजन

औद्योगिक डिझाइन आणि आर्किटेक्चर

वैज्ञानिक व्हिज्युअलायझेशन

वैद्यकीय फील्ड

२:संवर्धित वास्तविकता अगुमेंटेड रिॲलिटीची (AR):, संगणक प्रोग्रामिंगमध्ये**,** उपयुक्त संगणक-व्युत्पन्न डेटासह प्रतिमा आच्छादित करून व्हिडिओ किंवा फोटोग्राफिक डिस्प्ले एकत्र करण्याची किंवा "वाढवण्याची" प्रक्रिया.वापरकर्त्याला त्याचे कार्य पूर्ण करण्यात मदत करण्यासाठी एआर सिस्टम वास्तविक आणि आभासी एकत्र

करते. स्मार्टफोन आणि टॅब्लेटसारख्या मोबाईल उपकरणांनी आपल्या हातात आणि खिशात सुपर कॉम्प्युटरची शक्ती ठेवली आहे. जर आपण जगभर विचार करत असाल, तर कदाचित एखाद्या हेरिटेज साइटला भेट द्यावी

रोमांचक वास्तवाचा एक वर्धित अनुभव आहे. ऑगमेंटेड रिॲलिटी (एआर) ची हीच कल्पना आहे.

अगुमेंटेड रिॲलिटीची सर्वोत्तम वर्तमान उदाहरणे

IKEA मोबाइल ॲप

Nintendo चे **Pokémon Go** ॲप

Google Pixel चे **Star Wars** स्टिकर्स.

डिस्ने कलरिंग बुक. ...

L'Oreal मेकअप ॲप. ...

.पिरॅमिड किंवा एक आकर्षक परदेशी शहर आम्ही याआधी कधीच गेलो नव्हतो, आम्हाला जे हवे आहे ते सामान्यत: आभासी वास्तव नसून आमच्या समोर दिसणाऱ्या रोमांचक वास्तवाचा एक वर्धित अनुभव आहे. ऑगमेंटेड रिॲलिटी (एआर) ची हीच कल्पना आहे.

यूएस आर्मी.

घटक – II (Unit –II)

रास्टर स्कॅन ग्राफिक्स (Raster Scan Graphics)

गुण (18)

विषय निष्पत्ती (Course Outcome): स्टँडर्ड अल्गोरिदम अंमलात आणण्या करिता आणि विविध ग्राफिक्स आकार काढण्या करिता सी प्रोग्राम चा वापर करा **.**

घटक निष्पत्ती (Unit Outcome):

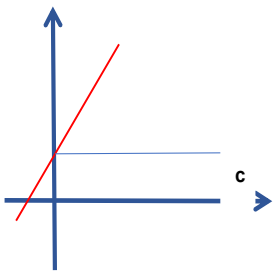
- दिलेल्या अल्गोरिदमचा वापर करून रेषा(लाइन) काढण्यासाठी प्रोग्राम लिहा.
- दिलेल्या रेषेला(लाइनला) रास्टराइज करण्यासाठी दिलेला अल्गोरिदम वापरा.
- वर्तुळ तयार करण्यासाठी दिलेला अल्गोरिदम लागू करा.
- दिलेल्या अल्गोरिदमचा वापर करून बहुभूज काढा.
- दिलेले कॅरॅक्टर प्रदर्शित करण्यासाठी कॅरॅक्टर निर्मिती पद्धत लागू करा.

२.१. लाईन रेखाटन अल्गोरिदम(Line Drawing Algorithms):**Basics Concepts of line Drawing**

पॉइंट अथवा बिंदू म्हणजे काय? (What is Point?)

पॉइंट किंवा पिक्सेल हे सर्वात लहान युनिट आहे ज्याला स्क्रीनवर संबोधित केले जाऊ शकते. पॉइंट हे दृश्याचे मूळ एकक आहे.

लाईन रेखाटन मूलभूत संकल्पना (Basic Concepts in line drawing) :



आकृती क्र. १ लाईन रेखाटन

y = किती अंतरा वर आहे (how far up)

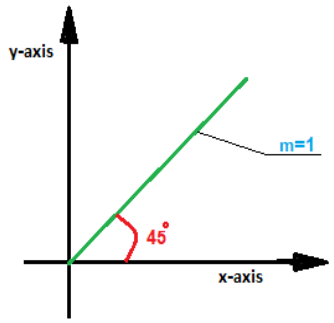
x = किती लांब आहे ?(how far along)

m = चढ किंवा ग्रेडियंट (Slope or Gradient) (how steep the line is)

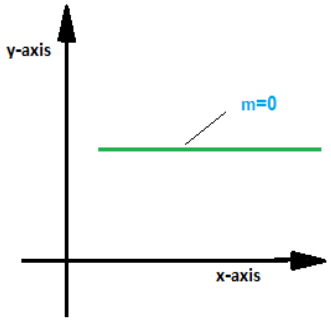
c = x=0 असताना y चे मूल्य (value of y when x=0**)**

तुम्ही "m" आणि "b" कसे शोधता ? (How do you find "m" and "b"?)

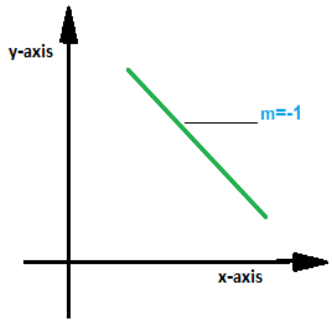
c लाइन Y अक्ष कुठे ओलांडते ते पहा (where the line crosses the Y axis).m चढ शोधा (calculate Slope).



आकृती क्र. २



आकृती क्र.३



आकृती क्र.४

$$m = \frac{y2 - y1}{x2 - x1}$$

२.१.१. लाईन रेखाटन अल्गोरिदम प्रकार(Types of Line Drawing Algorithm **) :**

१. डिजिटल डिफरन्शिअल ॲनालाईझर लाइन अल्गोरिदम DDA(Digital Differential Analyser)

२. ब्रेसेनहॅम लाइन अल्गोरिदम(Bresenham's Line algorithm).

२.१.२. डिजिटल डिफरन्शिअल ॲनालाईझर लाइन अल्गोरिदम DDA (Digital Differential Analyser):

- डीडीए व्हेरिएबल्सला(variables) एका बिंदूपासून दुस-या बिंदूपर्यंत इंटरपोलेट(interpolate) करण्यास मदत करते.
- डीडीए बहुभुज, लाइन आणि त्रिकोणांवर रास्टरायझेशन करते .
- डीडीएला स्कॅन रूपांतरणाची वाढीव पद्धत म्हणून देखील ओळखले जाते.
- या अल्गोरिदममध्ये, आपण टप्प्याटप्प्याने गणना करू शकतो.

आपल्याला माहित आहे की सरळ रेषेचे सामान्य समीकरण खालीलप्रमाणे :

$$y = mx + c$$

येथे, m हा (x1, y1) आणि (x2, y2) चा स्लोप slope) आहे.

म्हणजेच $m = \frac{y2-y1}{x2-x1}$

आता आपण एक बिंदू (x_k, y_k) आणि पुढील बिंदू (x_{k+1}, y_{k+1}) मानू.

$$m = \frac{(y_{k+1} - y_k)}{(x_{k+1} - x_k)}$$

आपल्याला प्रारंभ बिंदू आणि समाप्ती बिंदू दरम्यान स्लोप शोधायचा आहे. चर्चा करण्यासाठी खालील तीन प्रकरणे असू शकतात(३ cases):

Case 1: जर $m < 1$

$$x_{k+1} = x_k + 1$$

$$y_{k+1} = y_k + m$$

Case 2: जर $m > 1$

$$y_{k+1} = y_k + 1$$

$$x_{k+1} = x_k + \frac{1}{\mathrm{m}}$$

Case 3: जर $m = 1$

$$x_{k+1} = x_k + 1$$

$$y_{k+1} = y_k + 1$$

वरील तीन चर्चा केलेल्या प्रकरणांच्या मदतीने आपण सर्व मध्यवर्ती बिंदूंची गणना करू शकतो.

२.१.३. डिजिटल डिफरन्शिअल ॲनालाईझर लाइन अल्गोरिदम (Algorithm of Digital Differential Analyzer (DDA) Line Drawing):

Step 1: सुरू करा.

Step 2: आपण सुरवातीचा बिंदू (x1, y1) आणि शेवटचा बिंदू (x2, y2) मानूया

Step 3: dx आणि dy शोधायचे आहे
(calculate dx and dy.)

$dx = x_2 - x_1$

$dy = y_2 - y_1$

$m = dy/dx$

Step 4: तीन प्रकरणांशी तुलना करा (3 cases)

If $m < 1$

$(x_{k+1}, y_{k+1}) = (x_{k+1}, y_k + 1)$

If $m > 1$

$(x_{k+1}, y_{k+1}) = (x_{k+1}/m, y_{k+1}/m)$

If $m = 1$

$(x_{k+1}, y_{k+1}) = (x_{k+1}, y_k + 1)$

Step 5: जोपर्यंत आपल्याला रेषेचा शेवटचा बिंदू सापडत नाही तोपर्यंत आपण चरण 4 पुन्हा करू.

Step 6: थांबा (Stop).

२.१.४. डिजिटल डिफरन्शिअल ॲनालाईझर लाइन अल्गोरिदम चे फायदे(Advantages):

- हा अंमलात आणण्यासाठी एक सोपा अल्गोरिदम आहे.
- हा सरळ लाइन समीकरणापेक्षा वेगवान अल्गोरिदम आहे.
- डिजिटल डिफरेंशियल ॲनालायझरमध्ये आपण गुणाकार पद्धत वापरू शकत नाही.
- डिजिटल डिफरेंशियल ॲनालायझर अल्गोरिदम आपल्याला बिंदूच्या ओव्हरफ्लोबद्दल सांगतो, जेव्हा बिंदू त्याचे स्थान बदलतो.

डिजिटल डिफरन्शिअल ॲनालाईझर लाइन अल्गोरिदम तोटे(Disadvantages)**:**

- डिजिटल डिफरेंशियल ॲनालायझरची फ्लोटिंग पॉइंट अंकगणित अंमलबजावणी साठी जास्त वेळ लागतो .
- कधीकधी पॉइंट पोझिशन अचूक नसते.

२.१.५. उदाहरण :

रेषेचा प्रारंभ बिंदू (1,7) आणि शेवटचा बिंदू (11,17) आहे . लाइन प्लॉट करण्यासाठी डिजिटल डिफरेंशियल ॲनालायझर अल्गोरिदम लागू करा.

उदाहरण उत्तर :

आमच्याकडे दोन समन्वय आहेत, प्रारंभ बिंदू = (x1, y1) = (1,7) शेवटचा बिंदू = (x2, y2) = (11,17)

Step 1: प्रथम, आपण dx, dy आणि m ची गणना करतो.

$dx = x_2 - x_1 = 11\text{-}1 = 10$

$dy = y_2 - y_1 = 17\text{-}7 = 10$

$m = dy/dx = 10/10 = 1$

Step 2: आता, आपण चरणांची संख्या मोजू.

$dx = dy = 10$

त्यानंतर, चरणांची संख्या = 10

Step 3: m = 1 नुसार , तिसरी प्रकरण लागू होईल .

आता पुढील चरणावर जा.

x_k	y_k	x_{k+1}	y_{k+1}	(x_{k+1}, y_{k+1})
1	7	2	8	(2, 8)
		3	9	(3, 9)
		4	10	(4, 1)

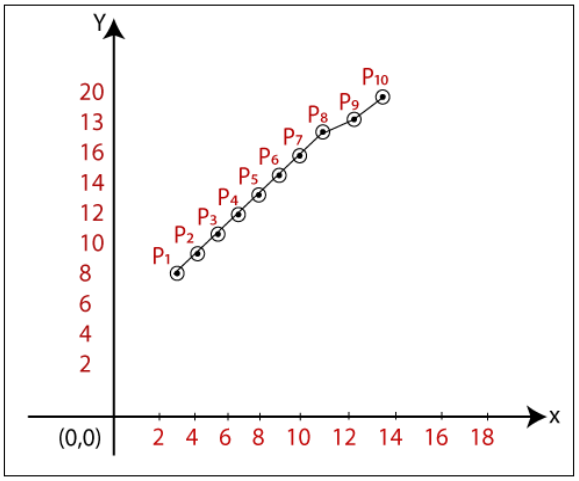
		5	11	(5, 11)
		6	1	(6, 12)
		7	13	(7, 13)
		8	14	(8, 14)
		9	15	(9, 15)
		10	16	(10, 16)
		11	17	(11, 17)

Step 4: जो पर्यंत आपल्याला रेषेचा शेवटचा बिंदू मिळत नाही तोपर्यंत आपण चरण 3 पुन्हा करू.

Step 5: थांबा.

काढलेल्या रेषेचे निर्देशांक आहेत:

P1 = (2, 8)
P2 = (3, 9)
P3 = (4, 10)
P4 = (5, 11)
P5 = (6, 12)
P6 = (7, 13)
P7 = (8, 14)
P8 = (9, 15)
P9 = (10, 16)
P10 = (11, 17)



आकृती क्र. ५ डीडीए लाइन अल्गोरिदम बिंदू ची काढणी

२.१.६. Program :// डीडीए लाइन अल्गोरिदम

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main()
{
int x1,y1,x2,y2,dx,dy,length,i;
float x,y,xinc,yinc;
int gd=DETECT,gm;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
printf("Enter the starting coordinates");
scanf("%d%d",&x1,&y1);
printf("Enter the ending coordinates");
scanf("%d%d",&x2,&y2);
dx=x2-x1;
dy=y2-y1;
if(abs(dx)>abs(dy))
```

```
        length=abs(dx);
else
        length=abs(dy);
xinc=dx/(float)length;
yinc=dy/(float)length;
 x=x1;
 y=y1;
putpixel(x,y,10);
 for(i=0;i<length;i++)
 {
 x=x+xinc;
 y=y+yinc;
 putpixel(x,y,10);
 delay(10);
 }
 getch();
 closegraph();
 }
```

२.१.७. ब्रेसेनहॅम लाइन अल्गोरिदम(Bresenham's Line algorithm):

- हा अल्गोरिदम १९६२ मध्ये "जॅक एल्टन ब्रेसेनहॅम" ने सादर केला होता.
- हा अल्गोरिदम आपल्याला एका ओळीचे स्कॅन रूपांतरण करण्यास मदत करतो.
- ही एक शक्तिशाली, उपयुक्त आणि अचूक पद्धत आहे.
- हे लाइन काढण्यासाठी वाढीव पूर्णांक गणना वापरते. पूर्णांक गणनेमध्ये बेरीज, वजाबाकी आणि गुणाकार यांचा समावेश होतो.

ब्रेसेनहॅमच्या लाइन ड्रॉइंग अल्गोरिदममध्ये, आपल्याला प्रारंभ बिंदू आणि शेवटच्या बिंदूमधील स्लोप (m) मोजावा लागेल.

२.१.८. ब्रेसेनहॅम लाइन अल्गोरिदम (Bresenham's Line Algorithm):

Step1: अल्गोरिदम सुरू करा (Start Algorithm)

Step2: व्हेरिएबल घोषित करा x_1,x_2,y_1,y_2,d,i_1,i_2,dx,dy (Declare variable)

Step3: मूल्ये घ्या x_1,y_1,x_2,y_2 (Enter value of)

x1,y1 हे सुरुवातीचे बिंदू आहे

And x2,y2 हे शेवटचे बिंदू आहे

Step4: $dx = x_2-x_1$

$dy = y_2-y_1$

$i_1=2*dy$

$i_2=2*(dy-dx)$

$d=i_1-dx$

Step5: (x, y) हे सुरुवातीचा बिंदू आणि x_{end} हि सर्वात जास्त असणारी x चे मूल्य असे मानूया .

जर $dx < 0$

तर $x = x_2$

$y = y_2$

$x_{end}=x_1$

जर $dx > 0$

तर $x = x_1$

$y = y_1$

$x_{end}=x_2$

Step6: (x, y) निर्देशांकांवर बिंदू निर्माण करा. (Generate point at (x,y)coordinates.)

Step7: संपूर्ण लाइन तयार झाली आहे का ते तपासा

जर $x >= x_{end}$

Stop.

Step8: पुढील पिक्सेलच्या निर्देशांकची गणना करा (Calculate co-ordinates of the next pixel)

जर $d < 0$

तर $d = d + i_1$

जर $d \geq 0$

तर $d = d + i_2$

$y = y + 1$

Step9: $x = x + 1$

Step10: नवीन (x, y) निर्देशांकांचा एक बिंदू काढा(Draw a point of latest (x, y) coordinates)

Step11: स्टेप 7 वर जा (Go to step 7)

Step12: अल्गोरिदमचा शेवट (End of Algorithm)

२.१.९. उदाहरण: रेषेची सुरुवात आणि शेवटची स्थिती (1, 1) व (8, 5) आहेत. मध्यवर्ती बिंदू शोधा.

उत्तर: $x_1=1$ $y_1=1$

$x_2=8$ $y_2=5$

$dx = x_2 - x_1 = 8-1=7$

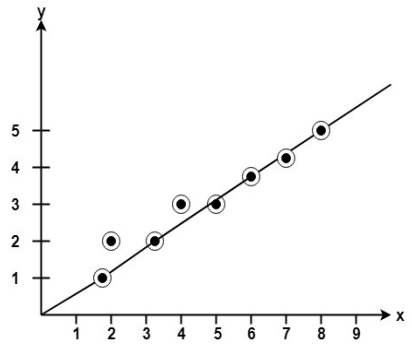
$dy = y_2 - y_1 = 5-1=4$

$I_1 = 2* dy = 2*4=8$

$I_2 = 2*(dy-dx) = 2*(4-7)=-6$

$d = I_1 - dx = 8-7=1$

x	Y	d=d+I1 or I2
1	1	d+I2=1+(-6)=-5
2	2	d+I1=-5+8=3
3	2	d+I2=3+(-6)=-3
4	3	d+I1=-3+8=5
5	3	d+I2=5+(-6)=-1
6	4	d+I1=-1+8=7
7	4	d+I2=7+(-6)=1
8	5	



२.१.१०. प्रोग्राम ब्रेसेनहॅम लाइन अल्गोरिदम

```
#include<stdio.h>
#include<graphics.h>
void drawline(int x0, int y0, int x1, int y1)
{
   int dx, dy, p, x, y;
   dx=x1-x0;
   dy=y1-y0;
   x=x0;
   y=y0;
   p=2*dy-dx;
   while(x<x1)
   {
      if(p>=0)
      {
         putpixel(x,y,7);
         y=y+1;
         p=p+2*dy-2*dx;
      }
      else
      {
         putpixel(x,y,7);
         p=p+2*dy;}
         x=x+1;
      }
}
```

```
int main()
{
    int gdriver=DETECT, gmode, error, x0, y0, x1, y1;
    initgraph(&gdriver, &gmode, "c:\\..\\bgi");
    printf("Enter co-ordinates of first point: ");
    scanf("%d%d", &x0, &y0);
    printf("Enter co-ordinates of second point: ");
    scanf("%d%d", &x1, &y1);
    drawline(x0, y0, x1, y1);
    return 0;
}
```

२.१.१२. डिजिटल डिफरन्शिअल ॲनालाईझर लाइन अल्गोरिदम आणि ब्रेसेनहॅम लाइन अल्गोरिदम यांच्यातील फरक:

अनु क्र.	**डिजिटल डिफरन्शिअल ॲनालाईझर लाइन अल्गोरिदम** DDA(Digital Differential Analyzer line)	**ब्रेसेनहॅम लाइन अल्गोरिदम**(Bresenham's Line algorithm).
१.	DDA अल्गोरिदम फ्लोटिंग पॉइंट वापरतो, म्हणजे वास्तविक अंकगणित.	ब्रेसेनहॅमचा लाइन अल्गोरिदम निश्चित बिंदू वापरतो, म्हणजे पूर्णांक अंकगणित.
२.	DDA अल्गोरिदम गुणाकार आणि भागाकार वापरते	ब्रेसेनहॅमचे लाइन अल्गोरिदम केवळ वजाबाकी आणि बेरीज वापरते
३.	DDA लाइन अल्गोरिदम ब्रेसेनहॅमच्या लाइन अल्गोरिदमपेक्षा हळू आहे कारण ते वास्तविक अंकगणित वापरते (फ्लोटिंग पॉइंट ऑपरेशन)	ब्रेसेनहॅमचा अल्गोरिदम हा DDA अल्गोरिदमपेक्षा वेगवान आहे कारण त्यात त्याच्या गणनेमध्ये फक्त बेरीज आणि वजाबाकी समाविष्ट आहे आणि फक्त पूर्णांक अंकगणित वापरतो.
४.	DDA अल्गोरिदम ब्रेसेनहॅमच्या लाइन अल्गोरिदमप्रमाणे अचूक आणि कार्यक्षम नाही.	Bresenham's Line Algorithm हा DDA अल्गोरिदमपेक्षा अधिक अचूक आणि कार्यक्षम आहे.
५.	DDA अल्गोरिदम वर्तुळ आणि वक्र काढू शकतो परंतु ब्रेसेनहॅमच्या लाइन अल्गोरिदमप्रमाणे अचूक नाही	ब्रेसेनहॅमचा लाइन अल्गोरिदम DDA अल्गोरिदमपेक्षा अधिक अचूक असलेले वर्तुळ आणि वक्र काढू शकतो.

२.१.१३. ब्रेसेनहॅमच्या लाइन ड्रॉइंग अल्गोरिदमचे फायदे:

- हे अंमलात आणणे सोपे आहे कारण त्यात फक्त पूर्णांक आहेत.

- ते जलद आणि वाढीव आहे.
- हे लागू करणे जलद आहे, परंतु डिजिटल डिफरेंशियल ॲनालायझर (DDA) अल्गोरिदमपेक्षा वेगवान नाही.
- पॉइंटिंग अचूकता DDA अल्गोरिदमपेक्षा जास्त आहे.

ब्रेसेनहॅमच्या लाइन ड्रॉइंग अल्गोरिदमचे तोटे:

- ब्रेसेनहॅमच्या लाइन ड्रॉइंग अल्गोरिदम केवळ मूळ लाइन काढण्यास मदत करते.
- परिणामी ड्रॉ लाइन स्मूथ काढू शकत नाही.

२.२. सर्कल जनरेटिंग अल्गोरिदम (Circle Generating Algorithm):

२.२.१. वर्तुळाची व्याख्या:

"सर्व बिंदू केंद्रबिंदूपासून समान अंतरावर (त्रिज्या) असतात अशा बिंदूंचे संयोजन म्हणून वर्तुळाची व्याख्या केली जाऊ शकते."

२.२.२. वर्तुळाची सममिती (Symmetry of Circle):

- वर्तुळ ही 8-बिंदू सममिती वर्तुळ (8-way symmetry)आकृती आहे. वर्तुळाचा आकार सर्व चतुर्थांशांमध्ये सारखाच असतो.
- प्रत्येक चतुर्थांशात दोन अष्टक असतात.
- एका अष्टकाच्या बिंदूची गणना केली तर 8-बिंदू सममिती(8-way symmetry) संकल्पना वापरून इतर सात बिंदू सहज काढता येतात.
- वर्तुळ ही एक भौमितिक आकृती आहे जी गोलाकार आहे आणि 360 अंशांमध्ये विभागली जाऊ शकते.
- वर्तुळ सममितीय आकृती जी 8-बिंदू सममितीचे(8-way symmetry) अनुसरण करते.

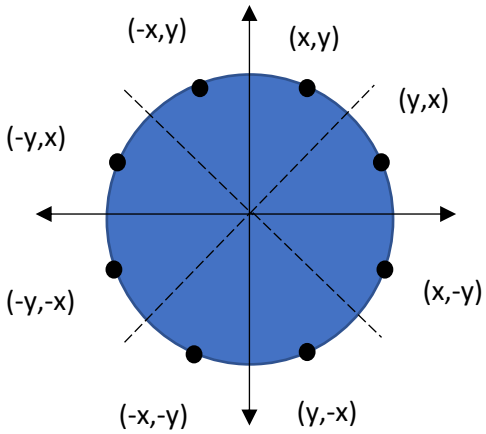
२.२.३. 8-**बिंदू सममिती**(8-way symmetry):

कोणतेही वर्तुळ 8-बिंदू सममितीचे अनुसरण करते. याचा अर्थ प्रत्येक वर्तुळासाठी (x,y) 8 बिंदू

प्लॉट केले जाऊ शकतात. हे (x,y), (y,x), (-y,x), (-x,y), (-x,-y), (-y,-x), (y,-x) (x,-y).

वर्तुळ ते मूळ मानते. जर बिंदू P1(x, y) असेल, तर इतर सात बिंदू असतील.

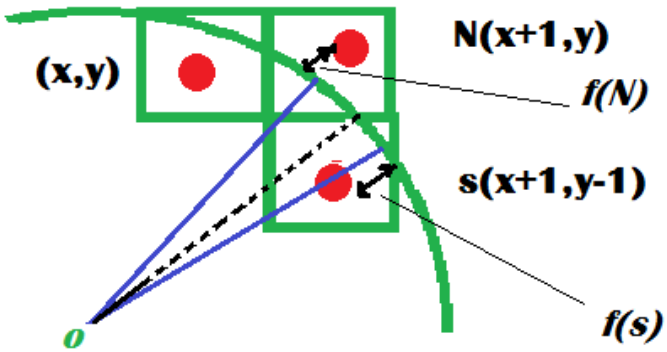
म्हणून आपण फक्त 45° परिघाचा भाग (arc) काढून . त्या वरून संपूर्ण वर्तुळ सहज ठरवता येईल.



आकृती क्र . ६ सर्कल

२.२.४. ब्रेसेनहॅम सर्कल अल्गोरिदम(Bresenham's Circle drawing algorithm):

- वर्तुळ काढण्यासाठी सर्वात कार्यक्षम आणि सर्वात सोपा अल्गोरिदम म्हणजे ब्रेसेनहॅम.
- सुरू करण्यासाठी, फक्त एक लक्षात ठेवा वर्तुळाचा अष्टांक तयार करणे आवश्यक आहे.
- इतर भाग क्रमिक प्रतिबिंबांद्वारे प्राप्त करू शकता,जर पहिला भाग (0 ते 45 ccw) शोधला तर त्यावरून इतर ऑक्टंट शोधता येतो .
- आपण रास्टर डिस्प्लेवर सतत आर्क प्रदर्शित करू शकत नाही. त्याऐवजी ८ भागचे आर्क पूर्ण करण्यासाठी आपल्याला सर्वात जवळच्या बिंदू ची निवड करावी लागेल.
- ब्रेसेनहॅमचे अल्गोरिदम हा मूळ-केंद्रित पहिल्या चतुर्थांशाचा विचार केला जातो . जर अल्गोरिदम x=0 , y=r पासून सुरू होत असेल, नंतर घड्याळाच्या दिशेने वर्तुळ y चे निर्माण होते .



आकृती क्र .७ ब्रेसेनहॅम सर्कल

२.२.५ ब्रेसेनहॅम सर्कल अल्गोरिदम (Algorithm) :

Step 1: त्रिज्या r व वर्तुळ केंद्र बिंदू (X_C, Y_C) स्वीकारा आणि पहिला बिंदू $(X_0,Y_0) = (0,r)$

Step 2: निर्णायक प्रारंभिक बिंदू शोधा $P_0= 3-2r$

Step 3: प्रत्येक वेळी X_K जागा ,सुरुवात k=0, ने करा ,खालील प्रमाणे पर्याय तपासा :

जर $P_k < 0$, नंतर चा बिंदू (X_{K+1},Y_K)

$$P_K+1=P_K+ 4X_K+6$$

अथवा सदर बिंदू (X_{K+1},Y_{K-1})

$$P_K+1=P_K+4(X_K-Y_K)+10$$

Step 4: इतर ७ बिंदू सममिती शोधा

Step 5: प्रत्येक बिंदू हलवा (X,Y) वर्तुळाकार पद्धतीने $X=X+X_C$ व $Y=Y+Y_C$

Step 6: पुन्हा step 3 to 5 पर्यंत X <= Y नाही .

२.२.६. उदाहरण :

वर्तुळ त्रिज्या r=10 दिल्यास, ब्रेसेनहॅम सर्कल अल्गोरिदम चा वापर करून X=0 पासून X=Y पर्यंत पहिल्या चतुर्थांशात वर्तुळचा भाग शोधा .

उत्तर:

प्रारंभिक निर्णय पॅरामीटर P_0.

$P_0=3-2r \quad =3-20 \quad = - 17$

$P_0 <0 \rightarrow (x1,y1)=(1,10)$

प्रारंभिक बिंदू आहे $(X_0, Y_0)=(0, 10)$ आणि प्रारंभिक वाढ हि अटी प्रमाणे होईल

1) $P_1=P_0+4*0+6 \quad = -17+ 0+ 6 \quad = -11$
2) $P_2=P_1+4*1+6 \quad = -11+ 4+ 6 \quad = -1$
3) $P_3=P_2+4*2+6 \quad = -1+ 8+ 6 \quad = 13$
4) $P_4=P_3+4(X3-Y3)+10 \quad = 13- 28+ 10 \quad = -5$
5) $P_5=P_4+4*4+6. \quad = -5+ 16+ 6 \quad = 17$
6) $P_6=P_5+4(X5-Y5)+10 \quad = 17- 16+ 10 \quad = 11$

k	P	X	Y	(X,Y)
0	-17	1	10	(1, 10)
1	-11	2	10	(2, 10)
2	-1	3	10	(3, 10)
3	13	4	9	(4, 9)
4	-5	5	9	(5, 9)
5	17	6	8	(6, 8)
6	11	7	7	(7, 7)

२.२.७. प्रोग्राम (Program) ब्रेसेनहॅम सर्कल अल्गोरिदम :

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<dos.h>
#include<graphics.h>
void bressn(int,int,int);
void plot(int,int,int,int);
void plot(int xc,int yc,int x,int y)
{
putpixel(xc+x,yc+y,WHITE);
putpixel(xc-x,yc+y,RED);
putpixel(xc-x,yc-y,GREEN);
putpixel(xc+x,yc-y,YELLOW);
putpixel(xc+y,yc+x,WHITE);
putpixel(xc-y,yc+x,GREEN);
putpixel(xc-y,yc-x,YELLOW);
putpixel(xc+y,yc-x,RED);
}
void bressn(int xc,int yc,int r)
{
```

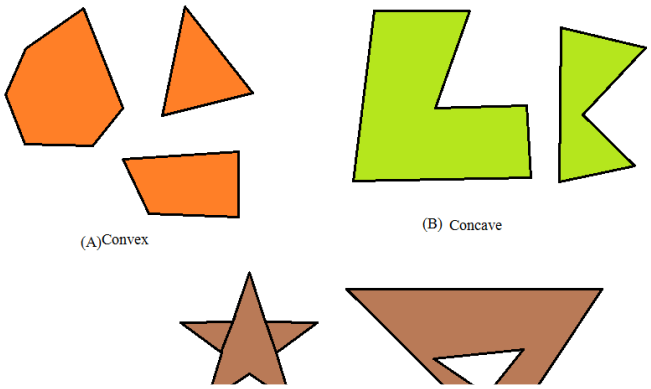
```
int d,x,y;
d=3-(2*r);
x=0;
y=r;
while(x<y)
{
if(d<0)
{
d=d+(4*x)+6;
}
else
{
d=d+4*(x-y)+10;
y=y-1;
}
x=x+1;
plot(xc,yc,x,y);
}
}
void main()
{
int gd,gm;
int xc,yc,radius;
detectgraph(&gd,&gm);
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
printf("\n enter xc,yc,radius");
scanf("%d%d%d",&xc,&yc,&radius);
bressn(xc,yc,radius);
getch();
closegraph();
}
```

२.३. बहुभुज(polygon)**:**

बहुभुज हा सरळ रेषांसह बनलेला कोणताही द्विमितीय(2-dimensional) आकार असतो. त्रिकोण, चतुर्भुज, पंचकोन आणि षटकोनी ही सर्व बहुभुजांची उदाहरणे आहेत.

२.३.१. बहुभुजाचे प्रकार(polygon Types**):**

- प्रत्येक बहुभुज एकतर उत्तल(convex) , अवतल (concave) आणि जटिल (complex).
- उत्तल आणि अवतल बहुभुजांमधील फरक त्यांच्या कोनांच्या मोजमापांमध्ये आहे.
- बहुभुज उत्तल होण्यासाठी, त्याचे सर्व आतील कोन 180^0 अंशांपेक्षा कमी असले पाहिजेत. अन्यथा, बहुभुज अवतल आहे.



आकृती क्र .८ बहुभुजाचे प्रकार

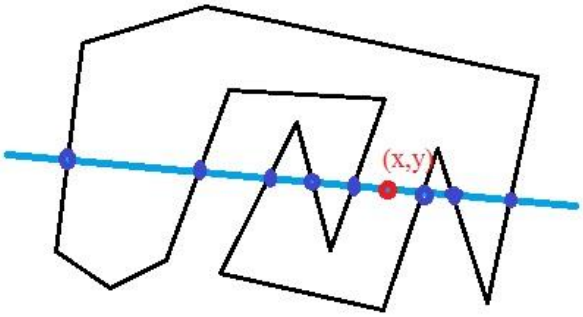
२.३.२. इन्साइड – आउटसाइड टेस्ट (Inside-Outside Test**):**

- ही पद्धत मोजणी संख्या पद्धत म्हणून देखील ओळखली जाते.
- विशिष्ट बिंदू क्षेत्रा च्या आत आहे की बाहेर आहे हे ओळखा.
- दोन पद्धती आहेत त्याद्वारे हे आपण ओळखू शकतो कि विशिष्ट बिंदू क्षेत्रा च्या आत आहे की बाहेर आहे.

1. विषम-सम नियम (Odd-Even Rule)
2. शून्य वळण संख्या नियम (Non zero winding number rule)

२.३.३. विषम-सम नियम (Odd-Even Rule**):**

या तंत्रात, आपण कोणत्याही बिंदूपासून (x,y) ते अनंतापर्यंतच्या रेषेच्या बाजूने किनारी(side) क्रॉसिंग मोजू. जर परस्परसंवादांची संख्या विषम आहे , तर बिंदू (x,y) हा आतील बिंदू आहे, आणि जर संख्या परस्परसंवादांची संख्या सम आहेत, तर बिंदू (x,y) हा बाह्य बिंदू आहे. खालील उदाहरण हि संकल्पना दर्शवते.

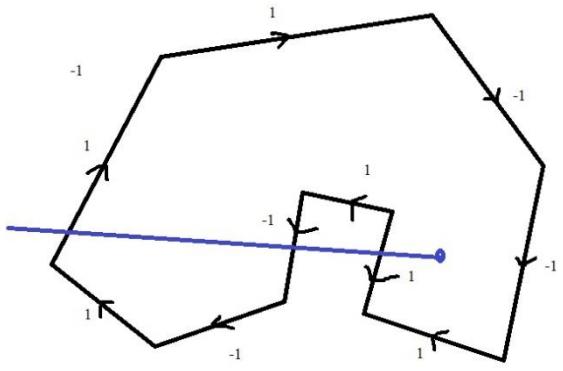


आकृती क्र .९ विषम-सम नियम

वरील आकृतीवरून, आपण बिंदू (x,y) वरून पाहू शकतो, वरील परस्परसंवाद बिंदूंची संख्या डावी बाजूस 5 आणि उजवीकडे 3 आहे. दोन्ही टोकांपासून, परस्परसंवाद बिंदूंची संख्या विषम आहे, म्हणून बिंदू त्या आकारात आहे असे विचारात घेतले जाते.

२.३.४. शून्य वळण संख्या नियम (Non zero winding number rule**):**

दिलेला बिंदू आतील आहे की नाही हे तपासण्यासाठी ही पद्धत साध्या बहुभुजा वर देखील वापरली जाते. वरच्या दिशेने जाणाऱ्या सर्व कडांना 1 आणि इतर सर्व -1 असे दिशा मूल्य द्या. किनारी दिशा मूल्ये तपासा ज्यामधून स्कॅन लाइन जात आहे आणि त्यांची बेरीज करा. या दिशेच्या मूल्याची एकूण बेरीज शून्य नसल्यास मग तपासले जाणारे बिंदू हे इंटीरियर बिंदू आहे. अन्यथा तो एक बाह्य बिंदू आहे.



आकृती क्र. १० शून्य वळण संख्या नियम

वरील आकृतीत स्कॅन लाइन ज्या दिशेतून जात आहे त्या दिशा मूल्यांची बेरीज करा एकूण 1 – 1 + 1 = 1 आहे. जे शून्य नाही . म्हणून त्या बिंदूला आंतरिक बिंदू म्हटले जाते.

२.३.५. बहुभुज भरणे(Polygon filling**):**

पॉलिगॉन फिलिंग अल्गोरिदमचे वर्गीकरण खालीलप्रमाणे केले आहे:

1. सीड फिल अल्गोरिदम (seed fill algorithm)
2. स्कॅन लाइन अल्गोरिदम(scan line algorithm)

बहुभूज भरण्या करिता दिलेल्या "सीड" बिंदू पासून सुरवात करून, जर तो बिंदू बहुभूजा च्या आतील बाजूस असेल तर शेजारील पिक्सेल जोपर्यंत आपल्याला बाउंडरी पिक्सेल येत नाही तो पर्यंत भरण्यात येते. या पद्धतीला सीड फिल म्हणतात.

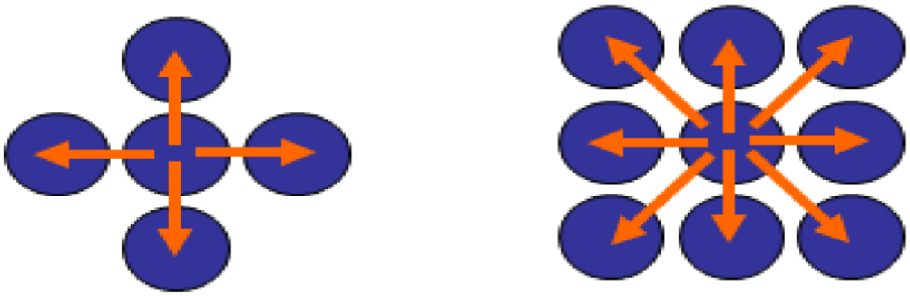
२.३.६.सीड फिल अल्गोरिदम प्रकार

1. फ्लड फिल (flood fill)
2. बाउंड्री फिल अल्गोरिदम (boundary fill)

अंतर्गत परिभाषित क्षेत्र भरणारे अल्गोरिदम म्हणजे फ्लड फिल अल्गोरिदम. ज्या बाउंड्रीच्या नुसार भरणारे क्षेत्र परिभाषित केल्या जातात त्यांना बाउंड्री फिल अल्गोरिदम म्हणतात. बहुभुज भरण्यासाठी आणखी एक दृष्टीकोन म्हणजे आतील चाचणी लागू करणे. पिक्सेल आहे की नाही हे तपासण्यासाठी.बहुभुजाच्या आत किंवा बहुभुजाच्या बाहेर आणि नंतर बहुभुजाच्या आत असलेले पिक्सेल हायलाइट करणे. हा दृष्टिकोन **स्कॅन-लाइन अल्गोरिदम** म्हणून ओळखला जातो.

२.३.७. फ्लड फिल अल्गोरिदम(Flood Fill Algorithm):

फ्लड फिल संपूर्ण क्षेत्राला एका संलग्न आकृतीमध्ये एक रंग वापरून परस्पर जोडलेल्या पिक्सेलद्वारे रंग देते. संगणक ग्राफिक्समध्ये रंग भरण्याचा हा एक सोपा मार्ग आहे. फक्त एक आकार घेतो आणि फ्लडफिल सुरू करतो. अल्गोरिदम अशा रीतीने कार्य करते जेणेकरुन सीमारेषे आतील सर्व पिक्सेलला समान रंग देता येईल.



आकृती क्र .११ 4–कनेक्ट आणि 8–कनेक्ट

1. बहुभुजाचा आतील भाग भरण्यासाठी फ्लड फिल अल्गोरिदम वापरला जातो.
2. एकापेक्षा जास्त रंगांच्या सीमांसह क्षेत्र परिभाषित केल्यावर वापरले जाते.
3. क्षेत्राच्या आतील एका बिंदूपासून प्रारंभ करा - एक निर्दिष्ट अंतर्गत रंग रंगाने बदला किंवा (जुना रंग) भरा

4. सर्व आतील बिंदू बदलले जाईपर्यंत 4–जोडलेला किंवा 8–जोडलेला प्रदेश भरा.

२.३.८. 4–कनेक्ट केलेले क्षेत्र दिलेल्या पिक्सेल वरून, तुम्ही 4 मार्गांनी काढा (उत्तर, दक्षिण, पूर्व,पश्चिम)

```
void fillcolor(int x,int y,int old_color,int new_color)
{
 if(getpixel(x,y)==old_color)
 {
 delay(5);
putpixel(x,y,new_color);
fillcolor(x+1,y,old_color,new_color);
fillcolor(x-1,y,old_color,new_color);
fillcolor(x,y+1,old_color,new_color);
fillcolor(x,y-1,old_color,new_color);
}
}
```

२.३.९. 8–कनेक्ट केलेले क्षेत्र दिलेल्या पिक्सेल वरून, तुम्ही 8-मार्गांच्या (उत्तर, दक्षिण, पूर्व, पश्चिम, ईशान्य, वायव्य, दक्षिणपूर्व, नैऋत्य) च्या मालिकेद्वारे ज्या क्षेत्रापर्यंत पोहोचू शकता

```
void fillcolor(int x,int y,int old_color,int new_color)
{
if(getpixel(x,y)==old_color)
{
delay(5);
putpixel(x,y,new_color);
fillcolor(x+1,y,old_color,new_color);
fillcolor(x-1,y,old_color,new_color);
fillcolor(x,y+1,old_color,new_color);
fillcolor(x,y-1,old_color,new_color);
fillcolor(x+1,y+1,old_color,new_color);
fillcolor(x+1,y-1,old_color,new_color);
fillcolor(x-1,y-1,old_color,new_color);
fillcolor(x-1,y+1,old_color,new_color);
```

```
}
}
```

२.३.१० फ्लड फिल चे फायदे(advantages) **:**

1. फ्लड फिल अल्गोरिदम सर्वात सोपा अल्गोरिदम आहे.

फ्लड फिल चे तोटे(disadvantages))**:**

2. फ्लड फिल अल्गोरिदम ला जास्त वेळ लागतो .

मोठ्या बहुभुज फिलसाठी फ्लड फिल अल्गोरिदम अयशस्वी आहे कारण त्याला मोठ्या फ्रेमची आवश्यकता आहे.

२.३.११. बाउंड्री फिल अल्गोरिदम(Boundary Fill Algorithm):

- आकृतीच्या आतील बाजूस प्रारंभ करतो आणि विशिष्ट रंगाने पेंट करतो . सीमेपर्यंत रंग भरतो .
- बाउंड्री फिल अल्गोरिदममध्ये संपूर्ण सीमा भरण्यासाठी रिकर्सिव्ह(recursive) पद्धत वापरली जाते.
- क्षेत्राच्या आतील एका बिंदूपासून प्रारंभ करतो .
- सीमेच्या दिशेने आतील बाजूस बाहेरून रंगवतो .
- सीमा एका रंगा ची असते.

२.३.१२. 4-**कनेक्टेड बाउंड्री फिल अल्गोरिदम** (4-connected boundary fill Algorithm):

```
boundary_fill(x, y, f_colour, b_colour)
{
if(getpixel(x, y)! = b_colour && getpixel(x, y)! = f_colour)
{
putpixel(x, y, f_colour);
boundary_fill(x +1, y, f_colour,b_colour);
boundary_fill(x, y+ 1,f_colour,b_colour);
boundary_fill(x -1, y, f_colour, b_colour);
boundary_fill(x, y – 1,f_colour,b_colour);
}
}
```

२.३.१३. 8-**कनेक्टेड बाउंड्री फिलअल्गोरिदम:**(8-connected boundary fillAlgorithm):

```
boundary_fill(x, y, f_colour, b_colour)
{
if(getpixel(x, y)! = b_colour && getpixel(x, y)! = f_colour)
{
```

```
putpixel(x, y, f_colour);
boundary_fill(x + 1, y, f_colour, b_colour);
boundary_fill(x - 1, y, f_colour, b_colour);
boundary_fill(x, y + 1, f_colour, b_colour);
boundary_fill(x, y - 1, f_colour, b_colour);
boundary_fill(x + 1, y + 1, f_colour, b_colour);
boundary_fill(x-1,y-1,f_colour,b_colour);
boundary_fill(x+1,y-1,f_colour,b_colour);
boundary_fill(x-1,y+1,f_colour,b_colour);
}
}
```

२.३.१४ . फ्लड फिल अल्गोरिदम व बाउंड्री फिल अल्गोरिदम यांच्या मधील फरक (Flood-fill Algorithm Vs Boundary-fill Algorithm**):**

तुलनेसाठी आधार	**फ्लड फिल अल्गोरिदम**	**बाउंड्री फिल अल्गोरिदम**
बेसिक	त्यात अनेक रंग असलेले क्षेत्र असू शकते.	हे केवळ एका रंगाने क्षेत्र परिभाषित करते.
पेंटिंग प्रक्रिया	आतील भाग रंगविण्यासाठी रॅनडम(random) रंगाचा वापर केला जाऊ शकतो, त्यानंतर जुना रंग नवीनसह बदलला जातो.	सीमेचा रंग सतत शोधून अंतर्गत बिंदू रंगवले जातात.
मेमरी चा वापर	जास्त आहे	कमी आहे
वेग	कमी आहे	जास्त आहे
अल्गोरिदम जटिलता	सोपा आहे	अवघड आहे

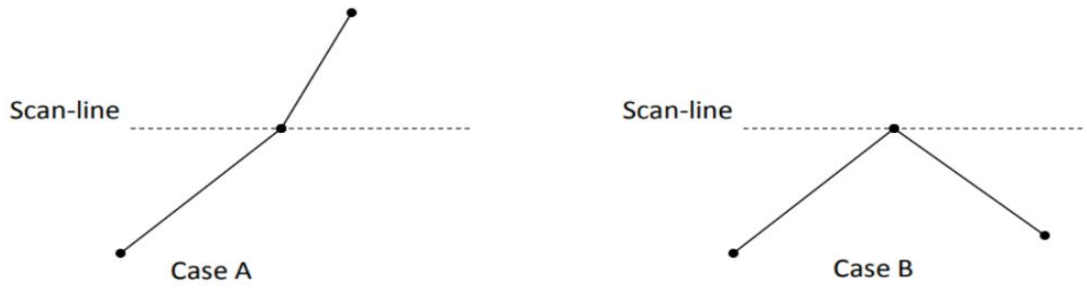
२.३.१५. स्कॅन लाइन फिल अल्गोरिदम (Scan Line Fill Algorithm**):**

बहुभुज यशस्वीरित्या भरण्यासाठी तीन मुख्य घटक वापरले जातात. एज टेबल(edge table) , एज बकेट (edge bucket) व सक्रीय यादी(active list) चा वापर केला जातो .

या घटकांमध्ये एजची माहिती असते , अनुक्रमे आकृती तयार करणाऱ्या सर्व कडा धरा आणि बहुभुज भरण्यासाठी वापरल्या जाणाऱ्या सध्याच्या कडा कायम ठेवा.जेव्हा स्कॅन लाइन एज च्या शेवटच्या पॉइंटला छेदते व ते दोन कडांना छेदते. तेव्हा खालील दोन प्रकरणांचा विचार करा :

प्रकरण (case) A: कडा नीरसपणे वाढत आहेत किंवा कमी होत आहेत. आपण हा फक्त एकच एज छेदनबिंदू मानले पाहिजे.

प्रकरण (case) B: शेवटच्या बिंदूवर कडा उलटी दिशेणी आहे का आणि आपण हे दोन एज छेदनबिंदू मानले पाहिजे.



आकृती क्र .१२ स्कॅन लाइन फिल अल्गोरिदम

२.४ . स्कॅन रूपांतरण (Scan conversion**):**

1. ग्राफिक्स तंत्रज्ञान हे अखंड ग्राफिक्स वस्तूंना स्वतंत्र पिक्सेलचा संग्रह म्हणून प्रस्तुत करण्याची प्रक्रिया आहे त्याला स्कॅन रूपांतरण म्हणतात.
2. स्कॅन रूपांतरण टीव्ही आणि संगणक यांच्यातील जोडण्याचे काम करते.
3. स्कॅन रूपांतरण किंवा स्कॅन रूपांतर रेट बदलण्यासाठी व्हिडिओ प्रक्रिया तंत्र आहे.
4. वेगवेगळ्या उद्देशांसाठी आणि अनुप्रयोगांसाठी व्हिडिओ सिग्नलची उभे /आडवे स्कॅन वापरतात.
5. जे उपकरण हे रूपांतरण करते त्याला स्कॅन कनवर्टर म्हणतात.
6. स्कॅन रूपांतरणाचा वापर : व्हिडिओप्रोजेक्टर, सिनेमा उपकरणे, टीव्ही आणि व्हिडिओ कॅप्चर कार्ड, मानक आणि एचडीटीव्ही टेलिव्हिजन, एलसीडी मॉनिटर्स, रडार डिस्प्ले आणि बरेच काही चित्र प्रक्रियेचे विविध पैलू.

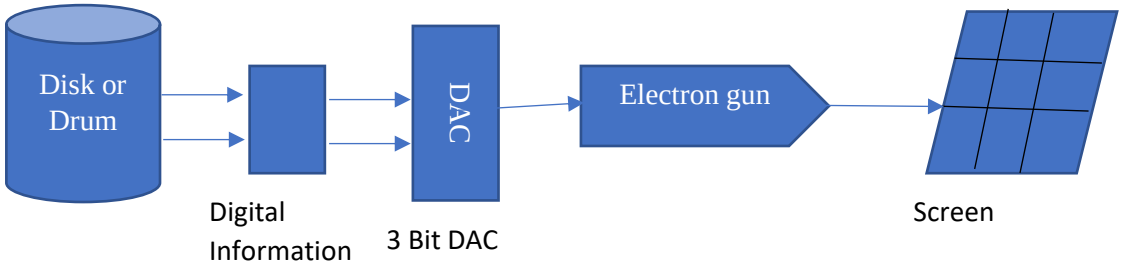
२.४.१. इमेज चा डेटा रेट बदलण्यासाठी दोन वेगळ्या पद्धती आहेत:

१. **ॲनालॉग पद्धती** (Analog Methods)**:**
(नॉन रिटेंटिव्ह, मेमरी-लेस किंवा रिअल टाइम पद्धत) हे रूपांतरण मोठ्या संख्येने विलंब सेल वापरून केले जाते आणि व्हिडिओ ॲनालॉगसाठी योग्य आहे. हे विशेष स्कॅन कन्व्हर्टर व्हॅक्यूमट्यूब वापरून देखील केले जाऊ शकते. या प्रकरणात ध्रुवीय समन्वय (कोन आणि अंतर) रडार सारख्या स्त्रोताकडून डेटा रिसीव्हर, जेणेकरून ते रास्टर स्कॅन (टीव्ही प्रकार) डिस्प्लेवर प्रदर्शित केले जाऊ शकते.

२. **डिजिटल पद्धती** (Digital methods)**:**
(रिटेंटिव्ह किंवा बफर पद्धत) या पद्धतीत, n1 गती (डेटा रेट) सह चित्र एका ओळीत किंवा फ्रेम बफरमध्ये संग्रहित केले जाते आणि n2 वेगाने वाचले जाते.

२.४.२. फ्रेम बफर (Frame buffer**):**

प्रत्येक स्क्रीन पिक्सेल मेमरीमध्ये राहणाऱ्या 2D ॲरेमधील विशिष्ट एंट्रीशी संबंधित आहे. ह्या मेमरीला फ्रेम बफर किंवा बिट मॅप म्हणतात. फ्रेम बफरमधील पंक्तींची संख्या डिस्प्ले स्क्रीन वरील रास्टर लाइनच्या संख्येइतकी आहे.. या ॲरेमधील स्तंभांची संख्या प्रत्येक रास्टर लाइन वरील पिक्सेलच्या संख्येइतकी आहे. वेगळ्या वेगळ्या प्रकारच्या फ्रेम बफर हा संगणक मेमरीचा एक मोठा भाग आहे जो डिस्प्ले इमेज साठवण्यासाठी वापरला जातो. मेमरी साठी ड्रम, डिस्क किंवा आयसी - शिफ्ट रजिस्टर्स सारख्या फ्रेम बफरसाठी मेमरी वापरली जाते . डिस्क किंवा ड्रम मध्ये साठवलेली माहिती हि डिजिटल मध्ये आहे. त्यामुळे DAC वापरून ते analog मध्ये रूपांतरित करणे आवश्यक आहे आणि नंतर हा ॲनालॉग सिग्नल पिक्सेल तयार करण्यासाठी वापरला जातो.



आकृती क्र .१३ इमेज प्रोसेस

२.५. कॅरेक्टर जनरेशन (Character Generation):

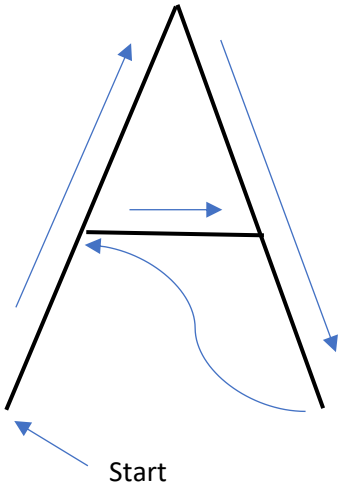
बऱ्याच वेळा अक्षरे ग्राफिक्स डिस्प्ले उपकरणांमध्ये तयार केली जातात, कधीकधी हार्डवेअर म्हणून वापरतात परंतु सॉफ्टवेअरद्वारे. मूलभूत तीन पद्धती आहेत:

1. स्ट्रोक पद्धत(stroke method)
2. स्टारबस्ट पद्धत(starburst method)
3. बिटमॅप पद्धत(bitmap method)

२.५.१. स्ट्रोक पद्धत(Stroke method):

ही पद्धत वर्ण तयार करण्यासाठी लहान रेषाखंड वापरते. रेषाखंडांच्या छोट्या मालिका पेनच्या स्ट्रोकप्रमाणे रेखाटल्या जातात ज्यामुळे एक वर्ण तयार होतो हे आकृतीत दाखवले आहे.

आपण स्वतःची स्ट्रोक पद्धत तयार करू शकतो.

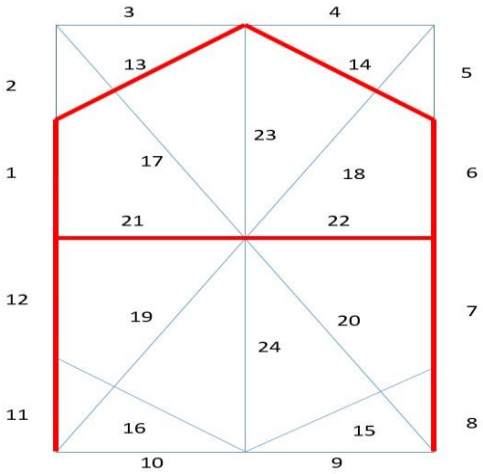


आकृती क्र .१४ स्ट्रोक पद्धत

1. लाइन ड्रॉइंग अल्गोरिदम कॉल करून.
2. येथे प्रत्येक वर्णासाठी कोणते रेषाखंड आवश्यक आहेत हे ठरविणे आवश्यक आहे.
3. नंतर लाइन ड्रॉइंग अल्गो वापरून हे सेगमेंट काढा.
4. ही पद्धत वर्ण स्केलिंगला समर्थन देते.
5. हे अक्षर रेखाटण्यासाठी वापरल्या जाणाऱ्या रेषाखंडांची लांबी बदलते .

२.५.२. स्टारबस्ट पद्धत(Starburst method):

1. पद्धतीमध्ये अक्षरे निर्माण करण्यासाठी रेषाखंडांचा एक निश्चित नमुना वापरला जातो
2. आकृतीत दाखवल्याप्रमाणे, 24 रेषाखंड आहेत
3. 24 ओळींच्या खंडांपैकी, विशिष्ट वर्णासाठी प्रदर्शित करण्यासाठी आवश्यक विभाग ठळक केले आहेत
4. या पद्धतीला तिच्या वैशिष्ट्यपूर्ण स्वरूपामुळे स्टारबस्ट पद्धत म्हणतात.



आकृती क्र .१५ स्टारबस्ट पद्धत

२.५.३. वर्ण निर्मितीच्या या पद्धतीचे काही तोटे आहेत ते खालील प्रमाणे :

1. अक्षराचे प्रतिनिधित्व करण्यासाठी 24-bit आवश्यक आहेत. त्यामुळे अधिक मेमोरी आवश्यक आहे.
2. त्याच्या 24-bit कोडमधून वर्ण प्रदर्शित करण्यासाठी कोड रूपांतरण सॉफ्टवेअर आवश्यक आहे.
3. वर्ण गुणवत्ता खराब आहे. वक्र आकाराच्या वर्णांसाठी ते सर्वात वाईट आहे.

२.५.४. बिटमॅप पद्धत(Bitmap method):

1. डॉट मॅट्रिक्स म्हणूनही ओळखले जाते कारण या पद्धतीमध्ये वर्ण अॅरे ठिपके मॅट्रिक्स स्वरूपात द्वारे दर्शविले जातात.
2. हा एक द्विमितीय(2d array) अॅरे आहे ज्यामध्ये स्तंभ आणि पंक्ती आहेत : आकृतीमध्ये दर्शविल्याप्रमाणे 5 X 7, 7 X 9 आणि 9 X 13 अॅरे(array) देखील वापरले जातात.
3. उच्च रिझोल्यूशन डिव्हाइसेस कॅरेक्टर अॅरे 100 X 100 वापरू शकतात.

1	1	1	1	0
0	1	0	0	1
0		0	0	1
0	1	1	1	0
0	1	0	0	1
0	1	0	0	1
1	1	1	1	0

आकृती क्र .१६ बिटमॅप

घटक 3 (Unit –III)

परिवर्तनांचा आढावा

(Overview Of Transformations) गुण (18)

विषय निष्पत्ती (Course Outcome):

- 2D आणि 3D ट्रान्सफॉर्मेशनसाठी प्रोग्राम लिहता येणे
- प्रोजेक्शन वापरून विएव प्लेन वर ओब्जेट बघता येणे

घटक निष्पत्ती (Unit Outcome):

- दिलेले ऑपरेशन 2D ट्रान्सफॉर्मेशनमध्ये करा.
- दिलेले ऑपरेशन 3D ट्रान्सफॉर्मेशनमध्ये करा.
- संमिश्र परिवर्तनावर आधारित समस्या सोडवा.
- ऑब्जेक्टवर प्रोजेक्शनचा प्रकार लागू करा.

कॉम्प्युटर ग्राफिक्स विविध कोनातून वस्तू पाहण्याची सुविधा देतात. रेखांकनासाठी संगणक वापरण्याचा उद्देश वापरकर्त्याला वस्तू वेगवेगळ्या कोनातून पाहण्याची सुविधा प्रदान करणे, वस्तूचे आकारमान किंवा आकार वाढवणे किंवा कमी करणे हा आहे ज्याला ट्रान्सफॉर्मेशन(transformation) म्हणतात.

परिवर्तन(transformation):Two dimensional Transformation:

परिवर्तन म्हणजे नियम लागू करून काही ग्राफिक्स बदलणे. आपण विविध प्रकारचे परिवर्तन करू शकतो जसे की, स्केलिंग ट्रान्सलेशन, रोटेशन, शिअरिंग इ. जेव्हा 2D प्लेन मध्ये परिवर्तन घडते तेव्हा त्याला 2D परिवर्तन (transformation) म्हणतात.

स्क्रीनवर ग्राफिक्सचे स्थान बदलण्यासाठी आणि त्यांचा आकार किंवा अभिमुखता बदलण्यासाठी संगणक ग्राफिक्समध्ये ट्रान्सफॉर्मेशन महत्त्वपूर्ण भूमिका बजावतात.

परिवर्तनाचे(transformation) दोन आवश्यक पैलू खाली दिले आहेत:

1. प्रत्येक परिवर्तनाचे (transformation) एकच अस्तित्व असते. हे अद्वितीय नाव किंवा चिन्हाद्वारे दर्शविले जाऊ शकते.
2. दोन परिवर्तने एकत्र करणे शक्य आहे, कनेक्ट केल्यानंतर एकच परिवर्तन प्राप्त होते, उदा. अनुवादासाठी A हे स्थानांतर परिवर्तन आहे. बी परिवर्तन स्केलिंग करते. दोनचे संयोजन C=AB आहे. तर C हा संकलित गुणधर्माद्वारे प्राप्त होतो.

3.1 परिवर्तनाचे प्रकार Types of transformations

1. स्थानांतर Translation
2. स्केलिंग Scaling
3. रोटेशन Rotation

4. प्रतिबिंब Reflection
5. शिअरिंग Shearing

3.1.1 स्थानांतर Translation

एखाद्या वस्तूच्या(object) एका स्थानावरून दुसऱ्या स्थितीत सरळ रेषेच्या हालचालीला ट्रान्सलेशन (Translation) म्हणतात. येथे वस्तू(object) एका समन्वय स्थानावरून (Coordinate Location)दुसऱ्या स्थानावर स्थित होतो.

बिंदूचे स्थानांतर (Translation of Point):

एका बिंदूचे समन्वय स्थान(Coordinate Location) (x, y) वरून दुसऱ्या (x1 y1) मध्ये स्थानांतर करण्यासाठी, स्थानांतर अंतर Tx आणि Ty मूळ समन्वयामध्ये जोडतो.

$$x1 = x + Tx \quad y1 = y + Ty$$

$$P' = P + T$$

$$\begin{bmatrix} x1 \\ y1 \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} Tx \\ Ty \end{bmatrix}$$

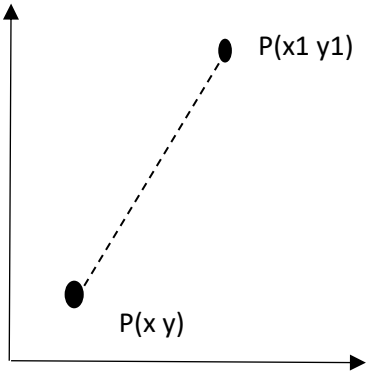
स्थानांतर जोडी (Tx,Ty) ला शिफ्ट वेक्टर(Shift Vector) म्हणतात.

स्थानांतर म्हणजे वस्तूचे(object) गुणधर्म न बदलता वस्तूंची (object)हालचाल. प्रत्येक बिंदू समान रकमेने रूपांतरित केला जातो.

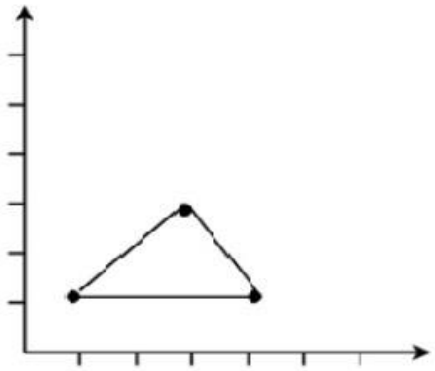
बहुभुजाचे(polygon) स्थानांतर करण्यासाठी, बहुभुजाचा प्रत्येक शिरोबिंदू(Vertex) नवीन स्थितीत रूपांतरित केला जातो.

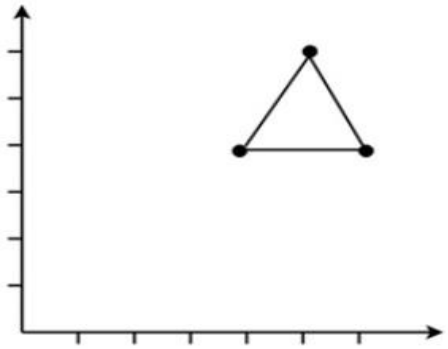
त्याचप्रमाणे, वर्तुळाची (Circle) किंवा लंबवर्तुळाकार (ellipse) स्थिती बदलण्यासाठी त्याचे केंद्र निर्देशांक(Center Coordinate) रूपांतरित केले जाते . नंतर नवीन निर्देशांक(Center Coordinate) वापरून वस्तू(object) रेखाटली जाते.

समजा P हा निर्देशांक (x, y) असलेला बिंदू आहे. त्याचे स्थानांतर (x1 y1) असे केले जाईल.



आकृती 3.1 बिंदूचे (Point) स्थानांतर





आकृती 3.2.1 बहुभुजाची मूळ स्थिती *आकृती 3.2.2 बहुभुजाची नवीन स्थिती*

Program: 2D त्रिकोणचे स्थानांतर Translation साठी प्रोग्राम

Program for 2D Translation of a Triangle.

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
#include<math.h>
int x1,y1,x2,y2,x3,y3,mx,my;
void drawtriangle();
void trans();
void main()
{
int gd=DETECT,gm; int c;
initgraph(&gd,&gm,"c:\\turboc3\\bgi ");
printf("Enter the 1st point for the triangle:");
scanf("%d%d",&x1,&y1);
printf("Enter the 2nd point for the triangle:");
scanf("%d%d",&x2,&y2);
printf("Enter the 3rd point for the triangle:");
scanf("%d%d",&x3,&y3);
cleardevice();
drawtriangle ();
getch();
trans();
getch();
}
void drawtriangle ()
{
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x3,y3,x1,y1);
}
void trans()
{
int x,y,a1,a2,a3,b1,b2,b3;
```

```
printf("Enter the Transaction coordinates");
scanf("%d%d",&x,&y);
cleardevice();
a1=x1+x;
b1=y1+y;
a2=x2+x;
b2=y2+y;
a3=x3+x;
b3=y3+y;
line(a1,b1,a2,b2);
line(a2,b2,a3,b3);
line(a3,b3,a1,b1);
}
```

3.1.2 Scaling स्केलिंग

स्केलिंग (Scaling) म्हणजे वस्तूंचा आकार बदलने. हा बदल दोन स्केलिंग घटक वापरून केला जातो. Sx म्हणजे x दिशेने ,Sy म्हणजे y दिशेने. मूळ स्थिती x आणि y असल्यास स्केलिंग घटक .Sx आणि Sy आहेत तर स्केलिंग नंतर निर्देशांकांचे मूल्य(Co-ordinates) (X', Y') आहेत

खाली दर्शविल्याप्रमाणे हे प्रस्तुत केले जाऊ शकते -

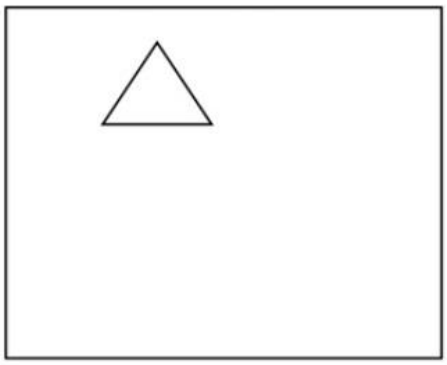
X' = X. SX आणि Y' = Y. SY

जर चित्र त्याच्या मूळ आकाराच्या दुप्पट मोठे करायचे असेल तर Sx = Sy =2 जर Sx and Sy समान नसेल तर स्केलिंग होईल परंतु ते चित्र लांबवेल किंवा विकृत होईल.स्केलिंग घटक (Sx,Sy) एकापेक्षा कमी असल्यास, ऑब्जेक्टचा आकार कमी होतो .स्केलिंग घटक (Sx,Sy) एकापेक्षा जास्त असल्यास, ऑब्जेक्टचा आकार मोठा होतो.Sx and Sy समान असल्यास त्याला एकसमान स्केलिंग (Uniform Scaling) असेही म्हणतात .समान नसल्यास डिफरन्शियल स्केलिंग (Differential Scaling) म्हणतात .एकापेक्षा कमी मूल्यांसह घटक स्केलिंग केल्यास ऑब्जेक्ट आरंभ निर्देशांकाच्या (coordinate origin) जवळ जाईल, तर एकापेक्षा जास्त असल्यास आरंभ निर्देशांकाच्या (coordinate origin)पासून दूर जाईल.

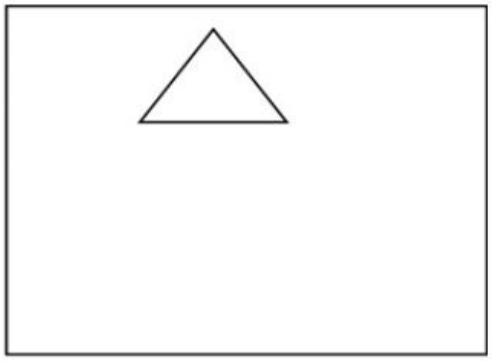
विस्तार **(Enlargement)**

$S=\begin{bmatrix}2 & 0\\ 0 & 2\end{bmatrix}$ आणि(x1 y1) मूळ निर्देशांक(Original Coordinates) असेल आणि S हे स्केलिंग मॅट्रिक्स(Scaling Matrix) असेल (x2 y2) हे स्केलिंगनंतरचे निर्देशांक (Coordinates) असतील तर

$$[x2\ y2]=[x1\ y1]\begin{bmatrix}2 & 0\\ 0 & 2\end{bmatrix}=[2x1\ 2y1]$$



आकृती 3.3.1 ओरीजनल इमेज



आकृती 3.3.2 नवीन इमेज

घट (Reduction) :

जर S = $\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$ आणि (x1,y1)मूळ निर्देशांक(Original Coordinates) असेल आणि S हे स्केलिंग मॅट्रिक्स(Scaling Matrix) असेल (x2,y2) हे स्केलिंगनंतरचे निर्देशांक (Coordinates) असतील तर

$$[x2\ y2]=[x1\ y1]\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}=[0.5x1\ ,0.5y1]$$

2D त्रिकोणचे स्केलिंग साठी प्रोग्राम Program for 2D scaling of triangle

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
#include<math.h>
int x1,y1,x2,y2,x3,y3,mx,my;
void drawtriangle();
void scale();
void main()
{
int gd=DETECT,gm;
int c;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
printf("Enter the 1st point for the triangle:");
scanf("%d%d",&x1,&y1);
printf("Enter the 2nd point for the triangle:");
scanf("%d%d",&x2,&y2);
printf("Enter the 3rd point for the triangle:");
scanf("%d%d",&x3,&y3);
drawtriangle ();
scale();
}
void drawtriangle ()
{
line(x1,y1,x2,y2);
```

```
line(x2,y2,x3,y3);
line(x3,y3,x1,y1);
}
void scale()
{
int x,y,a1,a2,a3,b1,b2,b3; int mx,my;
printf("Enter the scalling coordinates");
scanf("%d%d",&x,&y);
mx=(x1+x2+x3)/3;
my=(y1+y2+y3)/3;
cleardevice();
a1=mx+(x1-mx)*x;
b1=my+(y1-my)*y;
a2=mx+(x2-mx)*x;
b2=my+(y2-my)*y;
a3=mx+(x3-mx)*x;
b3=my+(y3-my)*y;
line(a1,b1,a2,b2);
line (a2,b2,a3,b3);
line(a3,b3,a1,b1);
drawtriangle ();
getch();
}
```

3.1.3 रोटेशन (Rotation)

वस्तूचा कोन(Angle) बदलण्याची प्रक्रिया म्हणजे रोटेशन (**Rotation)**.

रोटेशन घड्याळाच्या दिशेने(Clock Wise) किंवा विरुद्ध दिशेने (Anti Clockwise)असू शकते.

रोटेशनसाठी, आपल्याला रोटेशन कोन(Angle) आणि रोटेशन बिंदू (Rotation Point) घ्यावा लागेल.

रोटेशन बिंदूला(Rotation Point) पिव्होट बिंदू (Pivot point)देखील म्हणतात

रोटेशनचे प्रकार:

- घड्याळाच्या विरुद्ध दिशेने (Anticlockwise)
- घड्याळाच्या दिशेने(Clockwise)

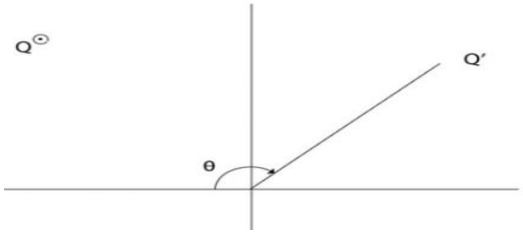
पिव्होट बिंदूचे मूल्य (Pivot point value) सकारात्मक(Positive) असेल तर वस्तूला (object)घड्याळाच्या उलट दिशेने फिरवते.पिव्होट बिंदूचे मूल्य (Pivot point value) निगेटिव्ह (Negative) असेल तर वस्तूला (object) घड्याळाच्या दिशेने फिरवते.जेव्हा वस्तूला (object) फिरवले जाते तेव्हा वस्तूचा(object) प्रत्येक बिंदू समान कोनाने (angle)फिरवला जातो.

सरळ रेषा(Striaght Line): सरळ रेषा(Line) शेवटच्या बिंदूंद्वारे(End Points) समान कोनातून(Angle) फिरवली जाते आणि नवीन शेवटच्या बिंदूंमधील(End points) रेषा पुन्हा रेखाटली जाते.

बहुभुज(Polygon): समान रोटेशनल कोन(angle) वापरून प्रत्येक शिरोबिंदू(Vertex) हलवून बहुभुज(Polygon) फिरवला जातो.

वक्र रेषा(Curved Lines): वक्र रेषा सर्व बिंदूंचे (points)स्थान बदलून आणि नवीन स्थानांवर वक्र (Curve)रेखाटून फिरवल्या जातात.

वर्तुळ(Circle): हे निर्दिष्ट कोनाद्वारे(Angle) केंद्रस्थानी(Center Point) मिळवता येते.

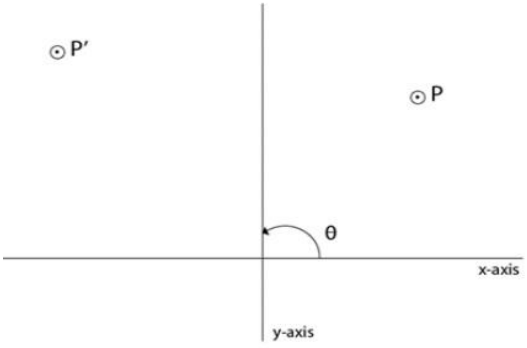


Q ही मूळ स्थिती आहे Q′ ही अंतिम फिरवलेली स्थिती आहे

आकृती 3.4 Q चे घड्याळाच्या दिशेने फिरणे

रोटेशन विथ रिस्पेक्ट टू ओरिजिन (Rotation with respect to origin)

- डीफॉल्ट रोटेशनसाठी सेंटर आरंभबिंदूं(Origin) (0,0) असते
- रोटेशन करण्यासाठी, रोटेशन कोन(Rotation Angle) θ द्यावा लागतो
- पिव्होट बिंदूं(pivot point) (xr , yr) द्यावा लागतो
- घड्याळाच्या उलट दिशेने(Anticlockwise) फिरण्यासाठी θ चे मूल्य(value) सकारात्मक असले पाहिजे
- घड्याळाच्या दिशेने फिरण्यासाठी θ चे मूल्य (value) नकारात्मक असले पाहिजे.

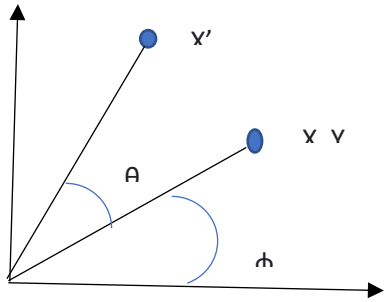


P मूळ स्थिती आहे

P′ हे अंतिम स्थान आहे

आकृती 3.5 *P घड्याळाच्या विरोधी दिशेने फिरणे*

(x,y)- ->रोटेशन विथ रिस्पेक्ट टु ओरिजिन बाय **θ(Rotation with respect to origin by θ)** --→(x′,y′)



आकृती 3.6 रोटेशन विथ रिस्पेक्ट टु ओरिजिन बाय θ**(Rotation with respect to origin)**

x = r cos (φ)

y = r sin (φ)

x′ = r cos (φ + θ)

y′ = r sin (φ + θ)

x′ = r cos (φ + θ)= r cos(φ) cos(θ) – r sin(φ) sin(θ)

= x cos(θ) – y sin(θ)

y′ = r sin (φ + θ) = r sin(φ) cos(θ) + r cos(φ)sin(θ)

= y cos(θ) + x sin(θ)

x′ = x cos(θ) – y sin(θ)

y′ = y cos(θ) + x sin(θ)

वरील समीकरण मॅट्रिक्स स्वरूपात खालील प्रमाणे होईल

$$[\mathbf{x}'y'] = [x\ y] \begin{bmatrix} \cos\theta & -\sin\theta \\ sin\theta & \cos\theta \end{bmatrix}$$

जेथे R हे रोटेशन मॅट्रिक्स (Rotation Matrix)आहे

$$P' = P.\ R$$

$$R=\begin{bmatrix} \cos\theta & -\sin\theta \\ sin\theta & \cos\theta \end{bmatrix}$$

सकारात्मक रोटेशन कोनासाठी(Positive Rotation Angle), आपण वरील रोटेशन मॅट्रिक्स (Rotation Matrix)वापरू शकतो. तथापि, नकारात्मक रोटेशन कोनासाठी, खाली दर्शविल्याप्रमाणे मॅट्रिक्स बदलेल

$$R=\begin{bmatrix} \cos(-\theta) & -\sin(-\theta) \\ sin(-\theta) & \cos(-\theta) \end{bmatrix}$$

cos(-θ)= cos(θ) आणि sin(-θ)=-sin(θ)

म्हणून

$$R=\begin{bmatrix} \cos\theta & \sin\theta \\ -sin\theta & \cos\theta \end{bmatrix}$$

2D त्रिकोणाच्या रोटेशन साठी प्रोग्राम Program for 2D Rotation of a Triangle

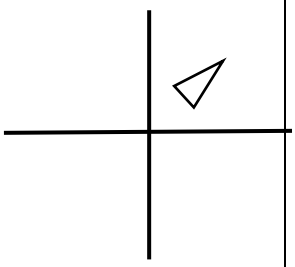
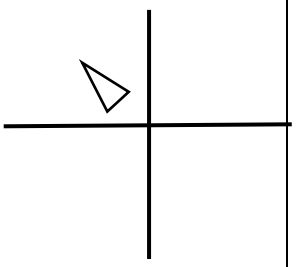
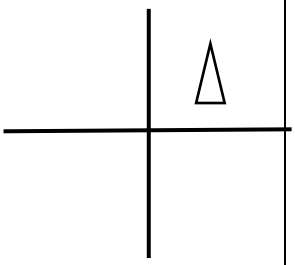
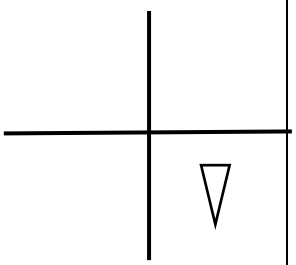
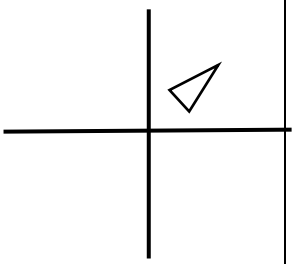
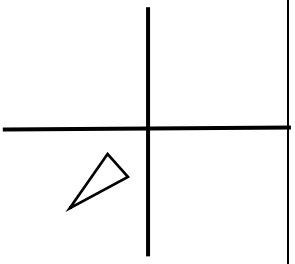
```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<process.h>
#include<math.h>
void triangle(int x1,int y1,int x2,int y2,int x3,int y3);
void Rotate(int x1,int y1,int x2,int y2,int x3,int y3);
void main()
{
int gd=DETECT,gm;
int x1,y1,x2,y2,x3,y3;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
printf("Enter the 1st point for the triangle:");
scanf("%d%d",&x1,&y1);
printf("Enter the 2nd point for the triangle:");
scanf("%d%d",&x2,&y2);
printf("Enter the 3rd point for the triangle:");
scanf("%d%d",&x3,&y3);
triangle(x1,y1,x2,y2,x3,y3);
getch();
cleardevice();
Rotate(x1,y1,x2,y2,x3,y3);
setcolor(1);
triangle(x1,y1,x2,y2,x3,y3);
getch();
}
void triangle(int x1,int y1,int x2,int y2,int x3,int y3)
{
line(x1,y1,x2,y2);
line(x2,y2,x3,y3);
line(x3,y3,x1,y1);
}
void Rotate(int x1,int y1,int x2,int y2,int x3,int y3)
{
int x,y,a1,b1,a2,b2,a3,b3,p=x2,q=y2;
float Angle;
printf("Enter the angle for rotation:");
scanf("%f",&Angle);
cleardevice();
Angle=(Angle*3.14)/180;
a1=p+(x1-p)*cos(Angle)-(y1-q)*sin(Angle);
b1=q+(x1-p)*sin(Angle)+(y1-q)*cos(Angle);
a2=p+(x2-p)*cos(Angle)-(y2-q)*sin(Angle);
b2=q+(x2-p)*sin(Angle)+(y2-q)*cos(Angle);
```

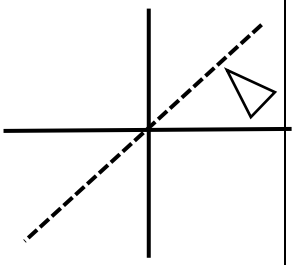
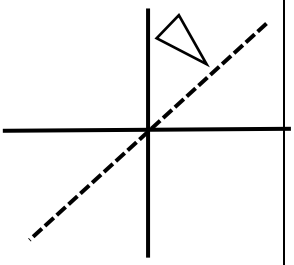
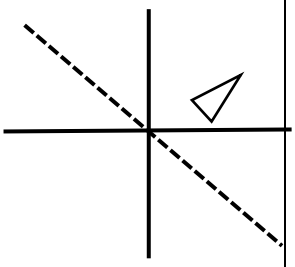
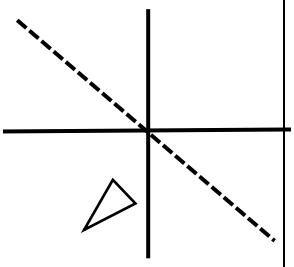
```
a3=p+(x3-p)*cos(Angle)-(y3-q)*sin(Angle);
b3=q+(x3-p)*sin(Angle)+(y3-q)*cos(Angle);
printf("Rotate");
triangle(a1,b1,a2,b2,a3,b3);
}
```

3.1.4 प्रतिबिंब (Reflection)

प्रतिबिंब ही मूळ वस्तूची(object) आरशातील प्रतिमा आहे. दुसऱ्या शब्दांत, आपण असे म्हणू शकतो की ही एक रोटेशनची कृती आहे . ज्याचा रोटेशन कोन (Rotation Angle) 180° आहे. प्रतिबिंब परिवर्तनांमध्ये वस्तूचा (object) आकार बदलत नाही.

खालील काही सामान्य प्रतिबिंबांची उदाहरणे आहेत:

Reflection प्रतिबिंब	**Transformation matrix** परिवर्तन मॅट्रिक्स	**Original image** मूळ प्रतिमा	**Reflected image** प्रतिबिंबित प्रतिमा
Reflection about Y axis Yअक्षा भोवती प्रतिबिंब	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$		
Reflection about X axis X अक्षा भोवती प्रतिबिंब	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$		
Reflection about origin ओरिजिन भोवती प्रतिबिंब	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$		

Reflection about line Y=X रेषा Y=X भोवती प्रतिबिंब	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$		
Reflection about line Y=-X रेषा Y=-X भोवती प्रतिबिंब	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$		

2D प्रतिबिंबासाठी प्रोग्राम Program to Perform 2D Reflection

```
#include<stdio.h>
#include<process.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
void disp(int n,float c[][3])
{
float maxx,maxy;
int i;
maxx=getmaxx();
maxy=getmaxy();
maxx=maxx/2;
maxy=maxy/2;
i=0;
while(i<n-1)
{
line(maxx+c[i][0],maxy-c[i][1],maxx+c[i+1][0],maxy-c[i+1][1]);
i++;
}
i=n-1;
line(maxx+c[i][0],maxy-c[i][1],maxx+c[0][0],maxy-c[0][1]);
setcolor(GREEN);
line(0,maxy,maxx*2,maxy);
line(maxx,0,maxx,maxy*2);
setcolor(WHITE);
}
```

```
void mul(int n,float b[][3],float c[][3],float a[][3])
{
int i,j,k;
for(i=0;i<n;i++)
for(j=0;j<3;j++)
a[i][j]=0;
for(i=0;i<n;i++)
for(j=0;j<3;j++)
for(k=0;k<3;k++)
{
a[i][j] = a[i][j] + (c[i][k] * b[k][j]);
}
}
void reflection(int n,float c[][3])
{
float b[10][3],a[10][3];
int i=0,ch,j;
cleardevice();
printf("\n\t* * MENU * *");
printf("\n\t1) ABOUT X-AXIS");
printf("\n\t2) ABOUT Y-AXIS");
printf("\n\t3) ABOUT ORIGIN");
printf("\n\t4) ABOUT Y=X");
printf("\n\t5) ABOUT Y=-X");
printf("\n\t6) EXIT"");
printf("\n\tENTER YOUR CHOICE :");
scanf("%d",&ch);
clrscr();
cleardevice();
disp(n,c);
for(i=0;i<3;i++)
for(j=0;j<3;j++)
{
b[i][j]=0;
if(i==j)
b[i][j]=1;
}
switch(ch)
{
case 1:
b[1][1]=-1;
break;
case 2:
b[0][0]=-1;
break;
case 3:
b[0][0]=-1;
```

```
b[1][1]=-1;
break;
case 4:
b[0][0]=0;
b[1][1]=0;
b[0][1]=1;
b[1][0]=1;
break;
case 5:
b[0][0]=0;
b[1][1]=0;
b[0][1]=-1;
b[1][0]=-1;
break;
case 6:
break;
default:
printf("”\n\tINVALID CHOICE ! ");
break;
}
mul(n,b,c,a);
setcolor(RED);
disp(n,a);
}
void main()
{
int i,j,k,cho,n,gd=DETECT,gm;
float c[10][3],tx,ty,sx,sy,ra;
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
printf("\nEnter the number of vertices : ");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("\nEnter the co-ordinates of the %d vertex :",i+1);
scanf("%f%f",&c[i][0],&c[i][1]);
c[i][2]=1;
}
disp(n,c);
reflection(n,c);
getch();
closegraph();
}
```

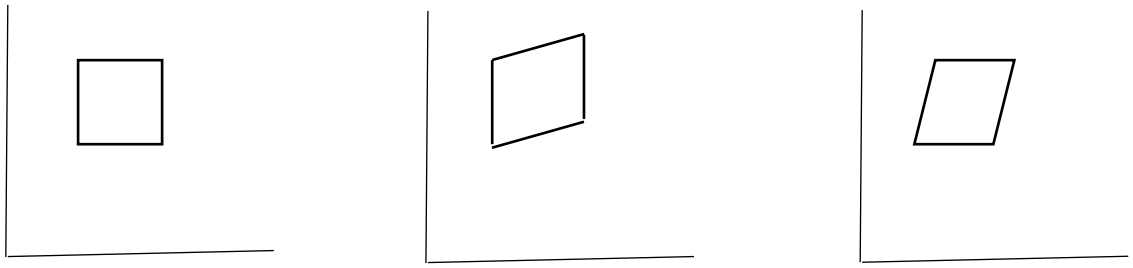
3.1.5 शिअरिंग (Shearing)

एखाद्या वस्तूच्या आकाराला तिरकस(Slant) बनवणाऱ्या परिवर्तनाला शिअर परिवर्तन (Shear Transformation)म्हणतात.

X-Shear आणि Y-Shear असे दोन शिअर परिवर्तन (Shear Transformation) आहेत.

X-Shear मध्ये X चे मूल्य(value) शिफ्ट होतात आणि

Y-shear मध्ये Y चे मूल्य (value) बदलतात.



ओरिजिनल वस्तू(Original object) Y Shear X Shear

आकृती 3.8 शिअरिंग परिवर्तन

X-Shear

X-Shear मध्ये Y मूल्य(value) राखून ठेवते आणि X मूल्यमध्ये(value) बदल केले जातात, ज्यामुळे उभ्या रेषा उजवीकडे किंवा डावीकडे झुकतात

X-Shear साठी परिवर्तन मॅट्रिक्स(Transformation Matrix) असे दर्शविले जाऊ शकते

$$Xsh=\begin{bmatrix}1 & 0\\ shx & 1\end{bmatrix}$$

$$X'= X+shx *Y$$

$$Y'=Y$$

Y Shear

Y-Shear मध्ये X मूल्य(value) राखून ठेवते आणि Y मूल्यमध्ये (value) बदल केले जातात. ज्यामुळे आडव्या रेषा(Horizontal Lines) वर किंवा खाली होतात.

Y-Shear साठी परिवर्तन मॅट्रिक्स(Transformation Matrix) असे दर्शविले जाऊ शकते

$$Ysh=\begin{bmatrix}1 & shy\\ 0 & 1\end{bmatrix}$$

$$Y'=Y+shy*X$$

$$X'=X$$

X आणि Y अक्षा भोवती शिअरिंग साठी प्रोग्राम Program for shearing of both X and Y Axis

```
#include<stdio.h>
#include<graphics.h>
#include<math.h>
int gd=DETECT,gm;
int n,xs[100],ys[100],i;
float shearXfactor,shearYfactor;
void DrawFun()
{
for(i=0;i<n;i++)
 line(xs[i],ys[i],xs[(i+1)%n],ys[(i+1)%n]);
}
void shearAlongX()
{
for(i=0;i<n;i++)
 xs[i]=xs[i]+shearXfactor*ys[i];
}
void shearAlongY()
{
for(i=0;i<n;i++)
 ys[i]=ys[i]+shearYfactor*xs[i];
}
void main()
{
printf("Enter number of sides: ");
scanf("%d",&n);
printf("Enter co-rdinates: x,y for each point ");
for(i=0;i<n;i++)
 scanf("%d%d",&xs[i],&ys[i]);
printf("Enter x shear factor:");
scanf("%f",&shearXfactor);
printf("Enter y shear factor:");
scanf("%f",&shearYfactor);
initgraph(&gd,&gm,"..\\BGI");
setcolor(RED);
DrawFun ();//original
shearAlongX();
setcolor(BLUE);
DrawFun ();                //Xshear
shearAlongY();
setcolor(GREEN);
DrawFun ();              //Yshear
getch();
}
```

3.2 एकसमान समन्वय Homogeneous coordinates:

चित्र घटकांना त्यांच्या योग्य स्थितीत बसवण्यासाठी डिझाईन आणि चित्र निर्मिती प्रक्रियेत, अनेक वेळा स्थानांतर, रोटेशन आणि स्केलिंग करावे लागते. या तीन परिवर्तनांना एकाच परिवर्तनामध्ये एकत्र करण्यासाठी, एकसमान निर्देशांक (Homogeneous coordinates)वापरले जातात. एकसमान निर्देशांक (Homogeneous coordinates) प्रणालीमध्ये, द्विमितीय समन्वय (two-dimensional coordinate) (x, y) तिहेरी-निर्देशांकांद्वारे triple-coordinates. दर्शविली जातात.

एकसमान निर्देशांक (Homogeneous coordinates) प्रणालीमध्ये समन्वय coordinates दर्शविण्याचे उदाहरण:

द्विमितीय भौमितिक परिवर्तनासाठी(two-dimensional geometric transformation), आपण शून्य नसलेल्या मूल्यासाठी h हा एकसमान(Homogeneous)पॅरामीटर निवडू शकतो. आपल्या सोयीसाठी h ची मूल्य(value) एक म्हणून घ्या.

प्रत्येक द्विमितीय स्थान नंतर एकसंध निर्देशांक (x, y, 1) सह दर्शविले जाते.

एकसमान निर्देशांकामध्ये (Homogeneous coordinates) द्विमितीय परिवर्तनासाठी खालील मॅट्रिक्स आहेत:

स्थानांतर (Translation) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ tx & ty & 1 \end{bmatrix}$ or $\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix}$

स्केलिंग (Scaling) $\begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$

घड्याळ्याच्या दिशेने रोटेशन (Clockwise Rotation) $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

घड्याळ्याच्या उलट दिशेने रोटेशन (Anticlockwise Rotation) $\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Xअक्षा भोवती प्रतिबिंब (Reflection against x-axis) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Yअक्षा भोवती प्रतिबिंब (Reflection against y-axis) $\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

आरंभबिंदू भोवती प्रतिबिंब(Reflection against origin) $\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

रेषा Y=X भोवती प्रतिबिंब (Reflection against line Y=X) $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

रेषा Y=-X भोवती प्रतिबिंब (Reflection against line Y=-X) $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Y दिशेने शिअरिंग (Shearing in x direction) $\begin{bmatrix} 1 & 0 & 0 \\ shx & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Y दिशेने शिअरिंग (Shearing in y direction) $\begin{bmatrix} 1 & shy & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

दोन्ही दिशेने शिअरिंग (Shearing in both direction) $\begin{bmatrix} 1 & shy & 0 \\ shx & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

3.3 संमिश्र परिवर्तन (Composite Transformation)

3.3.1 अनियंत्रित बिंदू भोवती रोटेशन (Rotation about arbitrary point)

अनेक परिवर्तने किंवा परिवर्तनांचा क्रम एका एकामध्ये एकत्र केला जाऊ शकतो ज्याला रचना (Composition)म्हणतात. परिणामी मॅट्रिक्सला संमिश्र मॅट्रिक्स(composite Matrix) म्हणतात. जोडण्याच्या प्रक्रियेला संयोग म्हणतात.

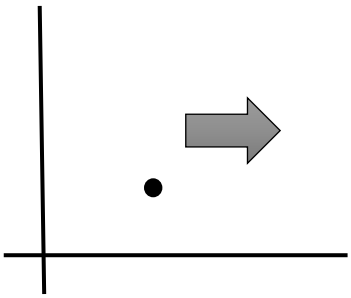
समजा आपल्याला एखाद्या वस्तूला(object) अनियंत्रित बिंदू (arbitrary point) भोवती रोटेट करायचा असेल तर, आपण तीन परिवर्तनांच्या(Transformation) खालील क्रमाने(Sequence)करू शकतो.

स्थानांतर (Translation)

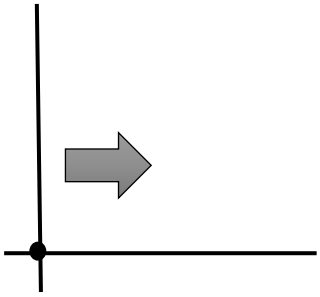
रोटेशन (Rotation)

उलट स्थानांतर (Reverse Translation)

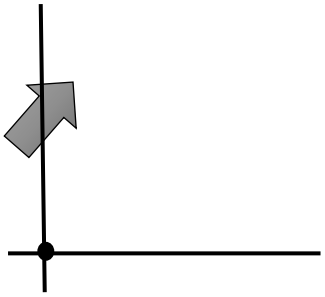
या परिवर्तनांचा क्रम बदलू नये.



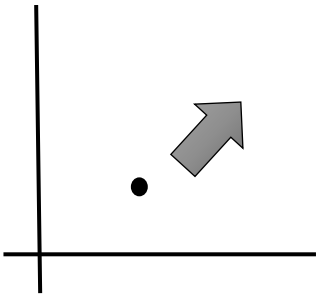
Step 1 ओरिजिनल वस्तू(object)



Step 2 अनियंत्रित बिंदूचे(arbitrary point)आरंभबिंदूला



Step 3 वस्तूचे आरंभबिंदू भोवती रोटेशन



Step 4 अनियंत्रित बिंदू (arbitrary point) चे पूर्वस्थानी स्थानांतर

आकृती 3.9 अनियंत्रित बिंदू भोवती रोटेशन (Rotation about arbitrary point)

उदाहरणार्थ, एखादी वस्तू एका अनियंत्रित बिंदूवर(Arbitory Point) (Xr, Yr) फिरवण्यासाठी, आपल्याला वर दिल्या प्रमाणे तीन कार्ये पार पाडावी लागतील.

Steps - (T1*R*T2)

(Xr, Yr) चे स्थानांतर आरंभबिंदूला(origin) करा. (Translate point (Xr, Yr) to the origin)

$$T1=\begin{bmatrix}1 & 0 & Xr\\ 0 & 1 & Yr\\ 0 & 0 & 1\end{bmatrix}$$

वस्तू (object)आरंभबिंदूला(origin) रोटेट करा. (Rotate it about the origin)

$$R=\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}$$

शेवटी, रोटेशनचे केंद्र जेथे होते तेथे स्थानांतर करा.(Finally, translate the center of rotation back where it belonged.)

$$T2=\begin{bmatrix}1 & 0 & -Xr\\ 0 & 1 & -Yr\\ 0 & 0 & 1\end{bmatrix}$$

P'=T1*R*T2

$$=\begin{bmatrix}1 & 0 & Xr\\ 0 & 1 & Yr\\ 0 & 0 & 1\end{bmatrix}*\begin{bmatrix}\cos\theta & -\sin\theta & 0\\ sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}*\begin{bmatrix}1 & 0 & -Xr\\ 0 & 1 & -Yr\\ 0 & 0 & 1\end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & Xr \\ 0 & 1 & Yr \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & -Xr\cos\theta + \mathrm{Yr}sin\theta \\ sin\theta & \cos\theta & -Xrsin\theta - \mathrm{Yr}cos\theta \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & Xr - Xr\cos\theta + \mathrm{Yr}sin\theta \\ sin\theta & \cos\theta & Yr - Xrsin\theta - \mathrm{Yr}cos\theta \\ 0 & 0 & 1 \end{bmatrix}$$

3.4 3D परिवर्तन **Three Dimensional Transformations**

3D परिवर्तन(3D Transformation) हे 2D परिवर्तनाचा(2D Transformation) विस्तार आहे. सगळे परिवर्तन (Transformation) स्पेस मध्ये न होता प्लेन मध्ये होतात

3.4.1 3D स्थानांतर(Translation)

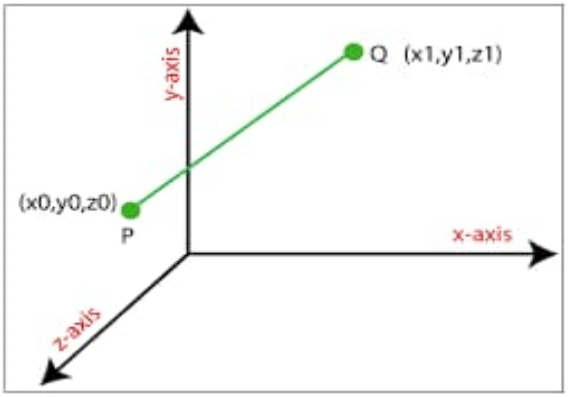
स्थानांतर म्हणजे एखाद्या वस्तूची एका स्थानावरून दुसऱ्या स्थितीत हालचाल होणे. स्थानांतर व्हेक्टर वापरून स्थानांतर केले जाते. 3D मध्ये दोन ऐवजी तीन वेक्टर आहेत. हे वेक्टर x, y आणि z दिशा मध्ये आहेत. X-दिशामधील स्थानांतर Tx वापरून दर्शविले जाते. स्थानांतर y-दिशा Ty वापरून दर्शविले जाते. z- दिशेतील स्थानांतर Tz वापरून दर्शविले जाते.

P हा एक बिंदू आहे ज्यामध्ये तीन दिशानिर्देश (Co-ordinates)आहेत (x, y, z), स्थानांतरानंतर त्याचे निर्देशांक (x1 y1 z1) असतील. Tx Ty Tz हे अनुक्रमे x, y, आणि z दिशानिर्देशांमधील स्थानांतर वेक्टर आहेत.

$$x^1 = x + T_x$$

$$y^1 = y + T_y$$

$$z^1 = z + T_z$$



आकृती 3.10 3D स्थानांतर

स्थानांतरचे मॅट्रिक्स(Matrix for Translation)

बिंदू (x, y, z). स्थानांतरानंतर(Translation) ते (x1, y 1,z1) बनते. Tx, Ty, Tz हे स्थानांतर वेक्टर (Translation Vector) आहेत.

$$\begin{bmatrix} x1 \\ y1 \\ z1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & Tx \\ 0 & 1 & 0 & Ty \\ 0 & 0 & 1 & Tz \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

3D स्थानांतरासाठी प्रोग्राम Program for 3D Translation

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<process.h>
#include<graphics.h>
int x1,x2,y1,y2,mx,my,depth;
void drawtranslation();
void translate();
void main()
{
int gd=DETECT,gm,c;
initgraph(&gd,&gm,"..\\BGI");
printf("\n\t\t3D Translation\n\n");
printf("\nEnter 1st top value(x1,y1):");
scanf("%d%d",&x1,&y1);
printf("Enter right bottom value(x2,y2):");
scanf("%d%d",&x2,&y2);
depth=(x2-x1)/4;
mx=(x1+x2)/2;
my=(y1+y2)/2;
drawtranslation ();
getch();
cleardevice();
translate ();
getch();
}
void drawtranslation ()
{
bar3d(x1,y1,x2,y2,depth,1);
}
void translate ()
{
int a1,a2,b1,b2,dep,x,y;
printf("\n Enter the Translation Distances:");
scanf("%d%d",&x,&y);
a1=x1+x;
```

```
a2=x2+x;
b1=y1+y;
b2=y2+y;
dep=(a2-a1)/4;
bar3d(a1,b1,a2,b2,dep,1);
setcolor(5);
drawtranslation ();
}
```

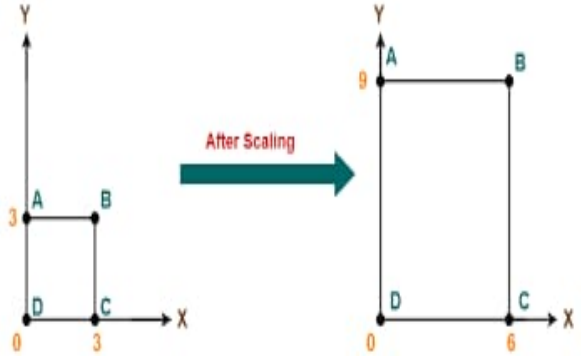
3.4.2 3D स्केलिंग 3D Scaling

स्केलिंगचा वापर वस्तूचा(object) आकार बदलण्यासाठी केला जातो. आकार वाढविला किंवा कमी केला जाऊ शकतो. स्केलिंग साठी तीन घटक आवश्यक आहेत Sx Sy आणि Sz.

S_x=x दिशेने स्केलिंग घटक (Scaling factor in x- direction)

S_y=y दिशेने स्केलिंग घटक (Scaling factor in y-direction)

S_z=z दिशेने स्केलिंग घटक (Scaling factor in z-direction)



आकृती 3.11 3D स्केलिंग(3D Scaling)

स्केलिंग करीता मॅट्रिक्स (Matrix for Scaling)

$$\begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D स्केलिंगसाठी प्रोग्राम Program for 3D Scaling

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<process.h>
#include<graphics.h>
int x1,x2,y1,y2,mx,my,depth;
void drawscaling();
void scaling();
void main()
{
int gd=DETECT,gm,c;
```

```
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
printf("\n\t\t3D Scaling\n\n");
printf("\nEnter 1st top value(x1,y1):");
scanf("%d%d",&x1,&y1);
printf("Enter right bottom value(x2,y2):");
scanf("%d%d",&x2,&y2);
depth=(x2-x1)/4;
mx=(x1+x2)/2;
my=(y1+y2)/2;
drawscaling ();
getch();
cleardevice();
scaling ();
getch();
}
void drawscaling ()
{
bar3d(x1,y1,x2,y2,depth,1);
}
void scaling ()
{
int x,y,a1,a2,b1,b2,dep;
printf("\n\n Enter scaling Factors:");
scanf("%d%d",&x,&y);
a1=mx+(x1-mx)*x;
a2=mx+(x2-mx)*x;
b1=my+(y1-my)*y;
b2=my+(y2-my)*y;
dep=(a2-a1)/4;
bar3d(a1,b1,a2,b2,dep,1);
setcolor(5);
drawscaling ();
}
```

3.4.3 3D Rotation

रोटेशन म्हणजे एखाद्या वस्तूला कोनामध्ये (angle) फिरवणे. 2D रोटेशनच्या तुलनेत 3D रोटेशन अवघड आहे. 2D रोटेशन साठी रोटेशनच्या कोनाचे(Rotation Angle)वर्णन करतो, परंतु 3D साठी रोटेशनचा कोन (Rotation Angle)आणि रोटेशनचा अक्ष(Rotation Axis) आवश्यक आहे. अक्ष (Axis) x किंवा y किंवा z असू शकतो

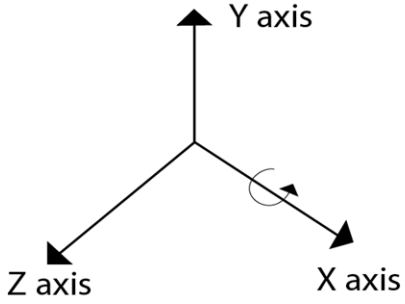
X अक्षा भोवती रोटेशन (Rotation about the X-axis)

या प्रकारच्या रोटेशनमध्ये, वस्तू x-अक्ष(axis) (मुख्य अक्ष) च्या समांतर फिरविली जाते, जेथे x समन्वय(coordinate) अपरिवर्तित राहतो आणि उर्वरित दोन समन्वय(coordinate) फक्त y आणि z बदलतात.

$$x' = x$$

$$y' = y\cos\theta - z\sin\theta$$

$$z' = y\sin\theta + z\cos\theta$$



आकृती 3.12 X अक्षा भोवती रोटेशन

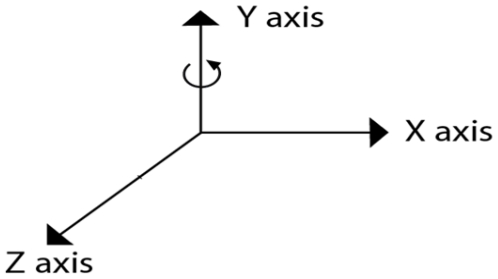
अक्षा भोवती Y रोटेशन(Rotation about the Y-axis)

या प्रकारच्या रोटेशनमध्ये, वस्तू Y-अक्ष(axis) (मुख्य अक्ष) च्या समांतर फिरविली जाते, जेथे Y समन्वय(coordinate) अपरिवर्तित राहतो आणि उर्वरित दोन समन्वय(coordinate) फक्त x आणि z बदलतात.

$$x' = z\sin\theta + x\cos\theta$$

$$y' = y$$

$$z' = z\cos\theta - x\sin\theta$$



आकृती 3.13 Yअक्षा भोवती रोटेशन

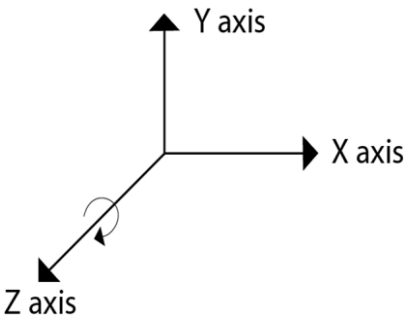
Z अक्षा भोवती रोटेशन (Rotation about the Z-axis)

या प्रकारच्या रोटेशनमध्ये, वस्तू Z-अक्ष(axis) (मुख्य अक्ष) च्या समांतर फिरविली जाते, जेथे z समन्वय(coordinate) अपरिवर्तित राहतो आणि उर्वरित दोन समन्वय(coordinate) फक्त x आणि y बदलतात.

$$x' = x \cos\theta - y \sin\theta$$

$$y' = x \sin\theta + y \cos\theta$$

$$z' = z$$



आकृती 3.14 Z अक्षा भोवती रोटेशन

3D रोटेशन साठी प्रोग्राम Program for 3D Rotation

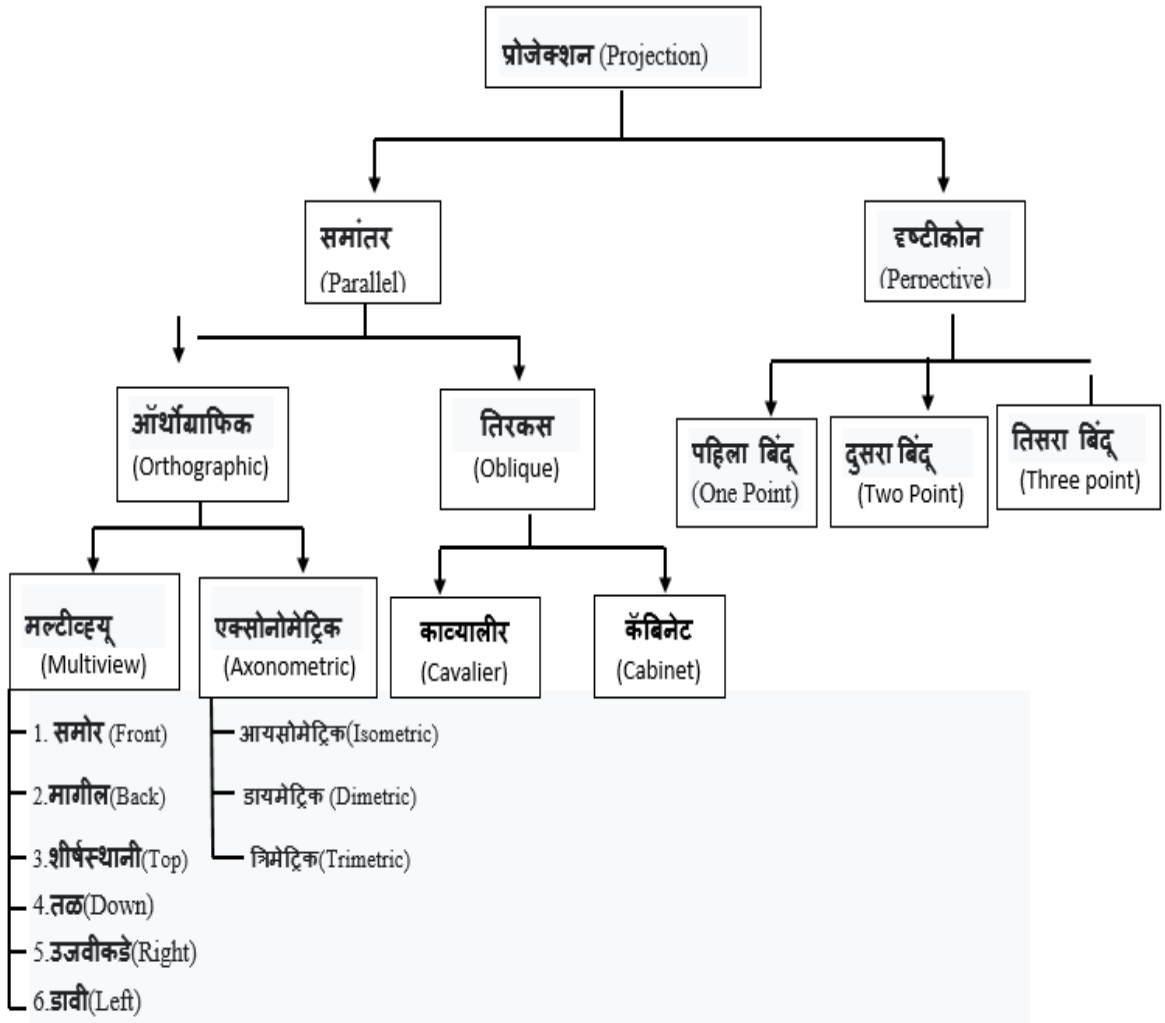
```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<math.h>
int maxx,maxy,midx,midy;
void axis()
{
getch();
cleardevice();
line(midx,0,midx,maxy);
line(0,midy,maxx,midy);
}
void main()
{
int x,y,z,o,x1,x2,y1,y2;
int gd=DETECT,gm;
detectgraph(&gd,&gm);
initgraph(&gd,&gm,"c:\\turboc3\\bgi");
maxx=getmaxx();
maxy=getmaxy();
midx=maxx/2;
midy=maxy/2;
axis();
bar3d(midx+50,midy-100,midx+60,midy-90,5,1);
printf("Enter rotating angle");
scanf("%d",&o);
x1=50*cos(o*3.14/180)-100*sin(o*3.14/180);
y1=50*sin(o*3.14/180)+100*cos(o*3.14/180);
x2=60*cos(o*3.14/180)-90*sin(o*3.14/180);
y2=60*sin(o*3.14/180)+90*cos(o*3.14/180);
```

```
axis();
printf("After rotation about z axis");
bar3d(midx+x1,midy-y1,midx+x2,midy-y2,5,1);
axis();
printf("After rotation about x axis");
bar3d(midx+50,midy-x1,midx+60,midy-x2,5,1);
axis();
printf("After rotation about yaxis");
bar3d(midx+x1,midy-100,midx+x2,midy-90,5,1);
getch();
closegraph();
}
```

3.5 प्रोजेक्शनचे प्रकार(Types of projection)

प्रोजेक्शनचा परिचय(Introduction to Projection)

प्रोजेक्शन हे 3D वस्तूला (object)2D वस्तूमध्ये रूपांतरित करण्याची प्रक्रिया आहे. प्रोजेक्शनला प्रोजेक्शन प्लेन(projection Plane) किंवा व्ह्यू प्लेनमध्ये(view Plane) ऑब्जेक्टचे मॅपिंग किंवा परिवर्तन(Transformation) म्हणून देखील परिभाषित केले जाते.



आकृती 3.15 प्रोजेक्शनचे प्रकार(Types of Projection)

3.5.1 समांतर प्रोजेक्शन Parallel Projection

चित्राचा खरा आकार दाखवण्यासाठी समांतर प्रोजेक्शनचा वापर केला जातो. जेव्हा प्रोजेक्टर(Projector) हा व्हियू प्लेन ला (View Plane) लंब (Perpendicular) असतो तेव्हा त्याला ऑर्थोग्राफिक प्रोजेक्शन (**orthographic projection**)म्हणतात.

समांतर प्रक्षेपण हा एक प्रकारचा प्रक्षेपण (projection)आहे जेथे प्रक्षेपण रेषा बहुभुज(polygon) पृष्ठभागावरून समांतरपणे बाहेर पडतात आणि नंतर समांतरपणे समतलपणे घडतात.

समांतर प्रोजेक्शन z-coordinate वापरत नाही व प्रत्येक शिरोबिंदू(vertex) पासून समांतर रेषा (Parallel Lines) वस्तू(object) व्ह्यू प्लेनला(View Plane) छेदत नाही तोपर्यंत विस्तारित केले जाते.

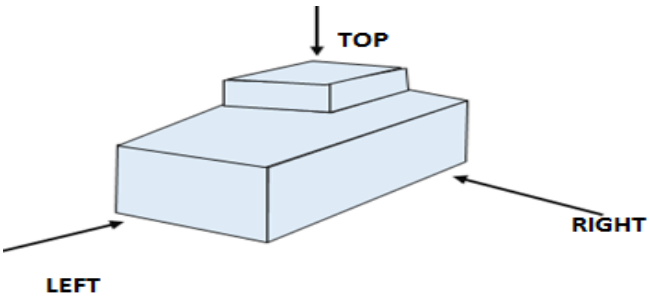
समांतर प्रोजेक्शनमध्ये, प्रक्षेपणाच्या केंद्राऐवजी प्रक्षेपणाची दिशा निर्दिष्ट करतो.

समांतर प्रोजेक्शन मध्ये, प्रक्षेपणाच्या केंद्रापासून प्रोजेक्ट प्लेनपर्यंतचे अंतर अगणित (infinite) असते.

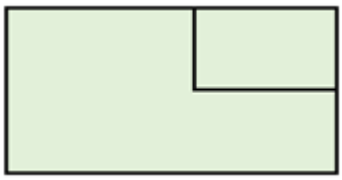
या प्रकारच्या प्रोजेक्शनमध्ये, आपण प्रक्षेपित शिरोबिंदूंना रेषाखंडांद्वारे जोडतो .

समांतर प्रोजेक्शनचा अंदाज कमी वास्तववादी आहेत, परंतु ते अचूक मोजमापांसाठी चांगले आहेत.

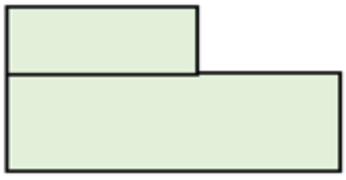
या प्रकारच्या प्रोजेक्शन मध्ये, समांतर रेषा(parallel Line) समांतर राहतात आणि कोन (Angle)जतन केले जात नाहीत.



आकृती 3.16 3D ऑब्जेक्ट



आकृती 3.16.1 वरून समांतर प्रक्षेपण



आकृती 3.16.2 डावीकडून समांतर प्रक्षेपण

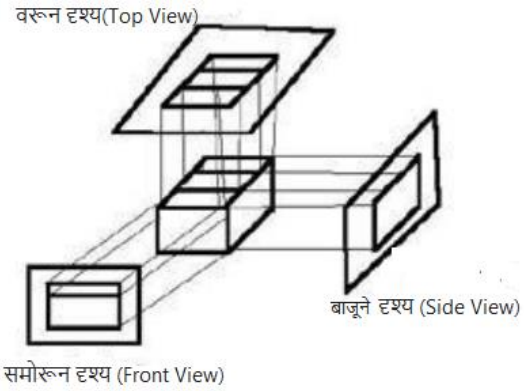
आकृती 3.16.3 उजवीकडून समांतर प्रक्षेपण

ऑर्थोग्राफिक प्रोजेक्शन (Orthographic Projection)

ऑर्थोग्राफिक प्रोजेक्शनमध्ये प्रक्षेपणाची दिशा प्रक्षेपणासाठी सामान्य पृष्ठभाग असते

ऑर्थोग्राफिक प्रोजेक्शनचे प्रकार

- फ्रंट प्रोजेक्शन (Front Projection)
- टॉप प्रोजेक्शन (Top Projection)
- साइड प्रोजेक्शन (Side Projection)



आकृती 3.17 ऑर्थोग्राफिक प्रोजेक्शन (Orthographic Projection)

तिरकस प्रोजेक्शन (Oblique Projection)

तिरकस प्रोजेक्शनमध्ये, (Oblique Projection) आपण ऑर्थोग्राफिक प्रोजेक्शनपेक्षा(Orthographic Projection) वस्तू अधिक चांगले पाहू शकतो.

दोन प्रकारचे तिरकस प्रोजेक्शन (Oblique Projection) आहेत –

कॅव्हलियर (Cavalier)

कॅबिनेट(Cabinet)

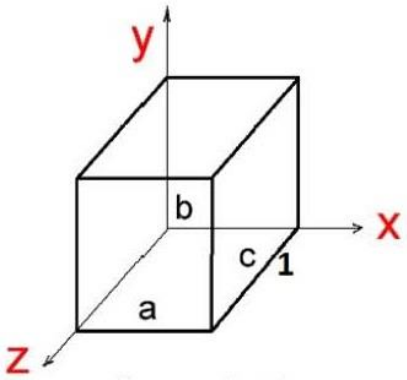
कॅव्हलियर प्रोजेक्शन प्रोजेक्शन प्लेन शी 45° कोन(Angle) बनवते.

कॅव्हलियर प्रोजेक्शनमध्ये व्ह्यू प्लेनला(view plane)लंब (perpendicular)असलेल्या रेषेच्या प्रक्षेपणाची(projection) लांबी स्वतःची रेषा सारखीच असते.

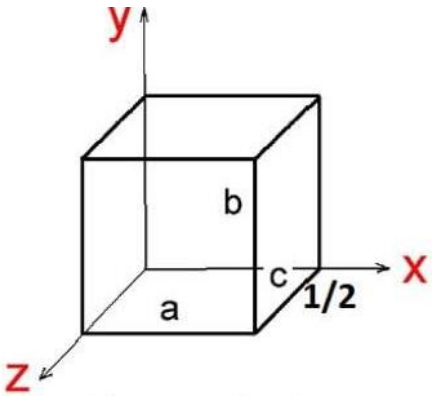
कॅव्हलियर प्रोजेक्शनमध्ये, पूर्वसंशोधन करणारे घटक तीनही प्रमुख दिशासाठी समान असतात

कॅबिनेट प्रोजेक्शन , प्रोजेक्शन प्लेनसह 63.4° कोन(Angle) बनवते.

कॅबिनेट प्रोजेक्शनमध्ये, पाहण्याच्या पृष्ठभागावर(Viewing Surface) लंब(perpendicular) असलेल्या रेषा त्यांच्या वास्तविक लांबीपेक्षा 1⁄2 ने प्रक्षेपित केल्या जातात.



आकृती 3.18 कॅव्हलियर प्रोजेक्शन Cavalier projection

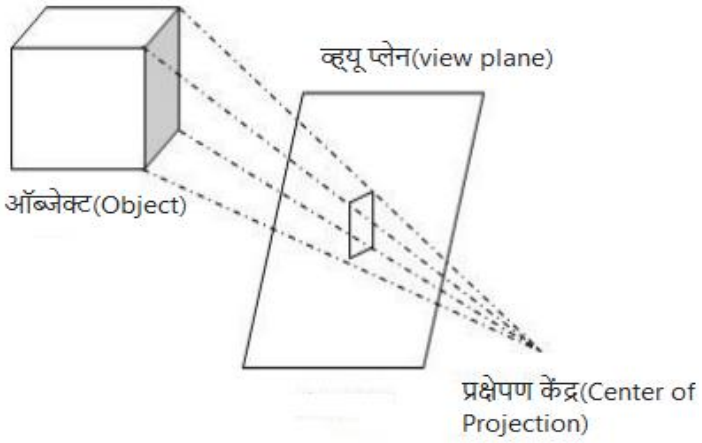


आकृती 3.19 कॅबिनेट प्रोजेक्शन Cabinet projection

3.5.2 दृष्टीकोन प्रोजेक्शन Perspective Projection

दृष्टीकोन प्रोजेक्शनमध्ये, प्रक्षेपणाच्या केंद्रापासून(Center of Projection) प्रोजेक्ट प्लेनपर्यंतचे अंतर मर्यादित असते.आणि वस्तूचा आकार अंतरानुसार उलट बदलतो ज्यामुळे ओब्जेट जास्त खरा(Real) वाटतो

अंतर(distance) आणि कोन (Angle) जतन केलेले नाहीत आणि समांतर रेषा(Parallel Lines) समांतर राहत नाहीत.त्याऐवजी, ते सर्व एकाच बिंदूवर एकत्र होतात ज्याला प्रक्षेपण(Projection) किंवा प्रक्षेपण केंद्र(Center of Projection) म्हणतात.



आकृती 3.20 दृष्टीकोन प्रोजेक्शन (Perspective Projection)

दृष्टीकोन प्रोजेक्शनचे खालील तीन प्रकार आहेत

एक बिंदू दृष्टीकोन प्रोजेक्शन(One point perspective projection)काढणे सोपे आहे.

दोन बिंदू दृष्टीकोन प्रोजेक्शन(Two point perspective projection)खोलीची चांगली छाप देते.

तीन बिंदू दृष्टीकोन प्रोजेक्शन(Three point perspective projection) काढणे सर्वात कठीण आहे.

समांतर प्रोजेक्शन आणि दृष्टीकोन प्रोजेक्शन मधील फरक

(Difference Between Parallel Projection and Perspective Projection

समांतर प्रोजेक्शन (Parallel Projection)	**दृष्टीकोन प्रोजेक्शन (Perspective Projection)**
समांतर प्रक्षेपण दुर्बिणीसारख्या वेगळ्या पद्धतीने वस्तूचे प्रतिनिधित्व करते. Parallel projection represents the object in a different way like telescope.	दृष्टीकोन प्रक्षेपण ऑब्जेक्टचे त्रिमितीय पद्धतीने(three dimensional) प्रतिनिधित्व करते. Perspective projection represents the object in three dimensional way.
समांतर प्रोजेक्शनमध्ये, हे प्रभाव तयार होत नाहीत. In parallel projection, these effects are not created.	दृष्टीकोन प्रक्षेपणात, दूर असलेल्या वस्तू लहान दिसतात आणि जवळच्या वस्तू मोठ्या दिसतात. In perspective projection, objects that are far away appear smaller, and objects that are near appear bigger
प्रक्षेपणाच्या केंद्रापासून वस्तूचे अंतर असंख्य (infinite) आहे. The distance of the object from the center of projection is infinite.	प्रक्षेपणाच्या केंद्रापासून वस्तूचे अंतर मर्यादित (finite)आहे. The distance of the object from the center of projection is finite.
समांतर प्रक्षेपण वस्तूचे(object) अचूक दृश्य देऊ शकते. Parallel projection can give the accurate view of object.	दृष्टीकोन प्रक्षेपण वस्तूचे(object) अचूक दृश्य देऊ शकत नाही Perspective projection cannot give the accurate view of object.
समांतर प्रक्षेपणाच्या रेषा समांतर(parallel) असतात The lines of parallel projection are parallel.	दृष्टीकोन प्रक्षेपणाच्या रेषा समांतर(parallel) नसतात The lines of perspective projection are not parallel.
समांतर प्रोजेक्शनमधील प्रोजेक्टर समांतर(parallel) असतात . Projector in parallel projection is parallel.	दृष्टीकोन प्रोजेक्शनमधील प्रोजेक्टर समांतर(parallel) नसतात Projector in perspective projection is not parallel.
समांतर प्रोजेक्शनचे प्रकार तिरकस प्रोजेक्शन (Oblique Projection) ऑर्थोग्राफिक Orthographic	दृष्टीकोन प्रोजेक्शनचे प्रकार एक बिंदू दृष्टीकोन प्रोजेक्शन(one point perspective)

Two types of parallel projection : 1.Orthographic, 2. Oblique Projection	दोन बिंदू दृष्टीकोन प्रोजेक्शन(two point perspective) तीन बिंदू दृष्टीकोन प्रोजेक्शन(three point perspective) Three types of perspective projection: 1.one point perspective, 2.Two point perspective, 3. Three point perspective,
वस्तूचे(object) वास्तववादी दृश्य तयार करत नाही It does not form realistic view of object	वस्तूचे(object) वास्तववादी दृश्य तयार करतात. It forms a realistic view of object.

घटक – ४ (Unit –IV)

विन्डोइंग आणि क्लिपिंग (Windowing and Clipping)

(Marks:14)

विषय निष्पत्ती (Course Outcome): विविध क्लिपिंग अल्गोरिदम लागू करा

घटक निष्पत्ती (Unit Outcome):

१. दिलेल्या ऑब्जेक्टवर विंडोव टू व्ह्यूपोर्ट ट्रान्सफॉर्मेशन लागू करा.
२. दिलेला लाइन क्लिपिंग अल्गोरिदम वापरून प्रोग्रॅम लिहा.
३. दिलेली लाइन क्लिप करण्यासाठी दिलेले क्लिपिंग अल्गोरिदम लागू करा.
४. दिलेल्या मजकुरावर मजकूर क्लिपिंग लागू करा..
५. दिलेला पॉलीगोन क्लिपिंग अल्गोरिदम वापरून प्रोग्राम लिहा.

४.१ विन्डोइंग आणि क्लिपिंग मूलभूत संकल्पना (Basic Concepts in Windowing and Clipping)

विन्डोइंग:(Windowing)

विंडो ही सिस्टीममधील कॉम्प्युटर डिस्प्ले स्क्रीनवर एक वेगळे पाहण्याचे क्षेत्र आहे जे ग्राफिकल यूजर इंटरफेस (GUI) चा भाग म्हणून एकाधिक दृश्य क्षेत्रांना अनुमती देते. विंडोज हे विंडोंग सिस्टमचा भाग म्हणून विंडोज मॅनेजरद्वारे व्यवस्थापित केले जाते. वापरकर्त्याद्वारे विंडोचा आकार बदलू शकतो.विंडो, प्रभावीपणे, आलेखाचा भाग परिभाषित करते जो जागतिक निर्देशांकांमध्ये प्रदर्शित केला जाणार आहे आणि व्ह्यूपोर्ट ज्या डिव्हाइसवर प्रतिमा दिसायची आहे ते क्षेत्र निर्दिष्ट करते.

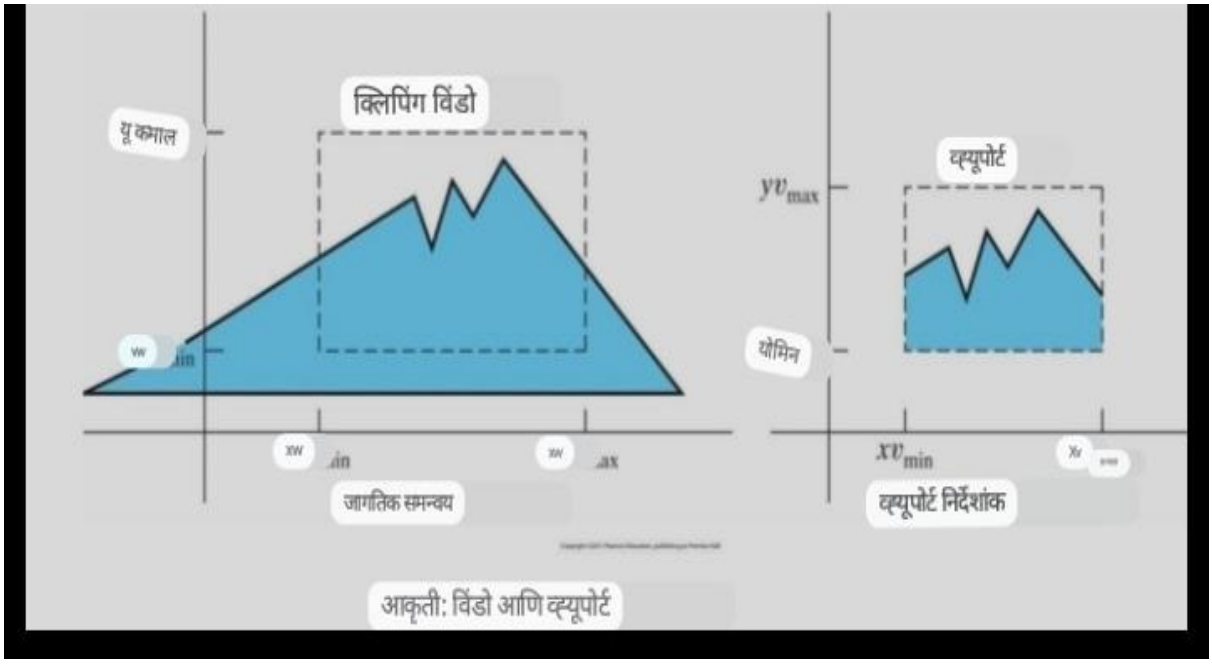
व्ह्यूपोर्ट:

व्ह्यूपोर्ट हा संगणक ग्राफिक्समधील बहुभुज पाहण्याचा प्रदेश आहे. व्ह्यूपोर्ट हे रेंडरिंग-डिव्हाइस-विशिष्ट निर्देशांकांमध्ये व्यक्त केलेले क्षेत्र आहे, उदा. स्क्रीन निर्देशांकांसाठी पिक्सेल, ज्यामध्ये स्वारस्य असलेल्या वस्तू प्रस्तुत केल्या जाणार आहेत.व्ह्यूपोर्ट डिस्प्ले डिव्हाइसवरील आयताकृती क्षेत्र सामान्यीकृत समन्वयामध्ये परिभाषित करते जेथे डेटाची प्रतिमा दिसते. तुम्ही GPORT कमांडसह व्ह्यूपोर्ट परिभाषित करता. तुम्ही तुमचा आलेख संपूर्ण डिस्प्ले डिव्हाइस घेऊ शकता किंवा तो फक्त एका भागामध्ये दाखवू शकता.

विंडोव टू व्ह्यूपोर्ट ट्रान्सफॉर्मेशन (Window to viewport Transformation)

विंडो **काय पहायचे** आहे ते परिभाषित करते; व्ह्यूपोर्ट ते कुठे असावे हे परिभाषित करते प्रदर्शित.

1. विंडो-टू-व्ह्यूपोर्ट ट्रान्सफॉर्मेशन ही द्विमितीय परिवर्तनाची प्रक्रिया आहे , जागतिक समन्वय देखावा करण्यासाठी साधन समन्वय विंडो-टू-व्ह्यूपोर्ट ट्रान्सफॉर्मेशन ही द्विमितीय, जागतिक-समन्वय दृश्याचे उपकरण समन्वयांमध्ये रूपांतर करण्याची प्रक्रिया आहे.
2. विशेषतः, जगातील वस्तू किंवा क्लिपिंग विंडो व्ह्यूपोर्टवर मॅप केल्या जातात. व्ह्यूपोर्ट स्क्रीनवरील इंटरफेस विंडोमध्ये प्रदर्शित होतो.
3. दुसऱ्या शब्दांत, क्लिपिंग विंडोचा वापर दृश्याचा भाग निवडण्यासाठी केला जातो जो प्रदर्शित केला जाणार आहे. व्ह्यूपोर्ट नंतर आउटपुट डिव्हाइसवर दृश्य ठेवतो.



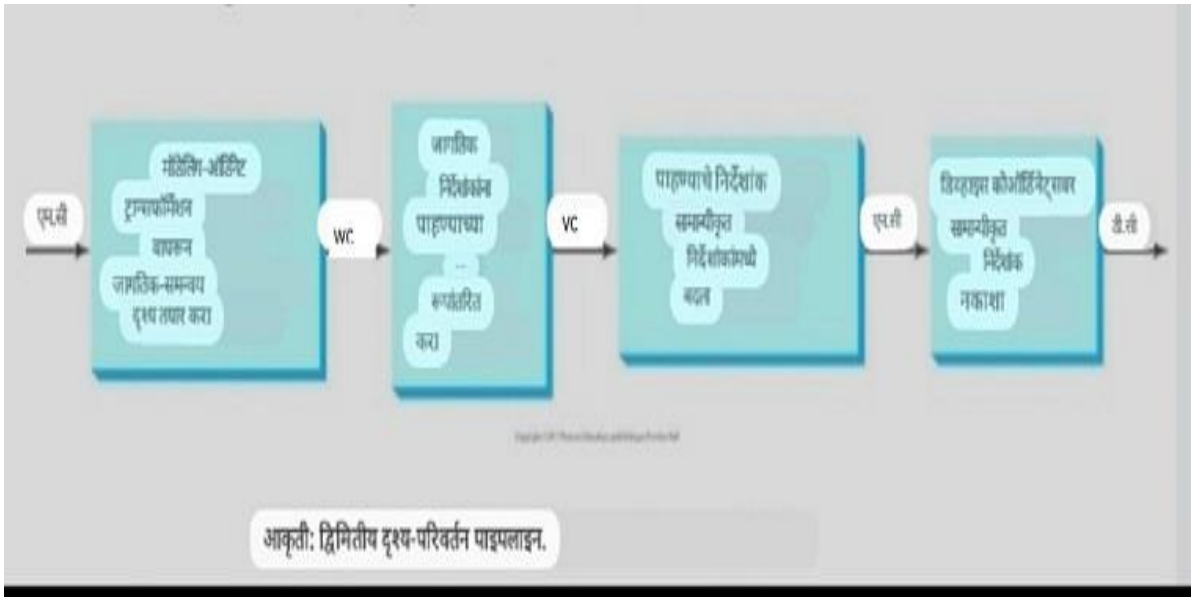
आकृती १ विंडोव आणि व्ह्यूपोर्ट

कोणत्याही सोयीस्कर कार्टेशियन को-ऑर्डिनेट प्रणालीचा वापर करून चित्र संगणकाच्या मेमरीमध्ये साठवले जाते, त्याला वर्ल्ड को-ऑर्डिनेट म्हणून संदर्भित प्रणाली (WCS) साम्भोदले जाते . तथापि, जेव्हा चित्र प्रदर्शित केले जाते प्रदर्शन साधनात डिस्प्ले डिव्हाइस मध्ये त्याला साधन को-ऑर्डिनेट करा प्रणाली (PDCS) म्हणतात.

- WC TO VC मध्ये रूपांतर .
- नंतर NC सामान्यीकरण पाहणे .
- NC TO DC मध्ये रूपांतर .

WC-विंडो समन्वय, NC: सामान्यीकरण समन्वय, VC-व्ह्यूपोर्ट समन्वय, DC: डिस्प्ले डिव्हाइस समन्वय

(WC-window coordinate, NC: normalize coordinate,VC-viewport coordinate,DC:display device coordinate)



आकृती: द्विमितीय दृश्य-परिवर्तन पाइपलाइन.

आकृती २ विंडोव टू व्ह्यूपोर्ट ट्रान्सफॉर्मेशन

परिवर्तन करण्याच्या पायऱ्या:-

१. आउटपुट आदिम(Primitive) आणि विशेषता वापरून जागतिक समन्वयामध्ये (WC)दृश्य तयार करा.
२. वर्ल्ड को-ऑर्डिनेट प्लेनमध्ये द्विमितीय दृश्य समन्वय प्रणाली सेट करून विंडोसाठी विशिष्ट अभिमुखता प्राप्त करा आणि दृश्य समन्वय प्रणालीमध्ये विंडो परिभाषित करा.
३. आयताकृती खिडक्यांसाठी अनियंत्रित अभिमुखता सेट करण्यासाठी पद्धत प्रदान करण्यासाठी को-ऑर्डिनेट्स संदर्भ फ्रेम पहा.
४. एकदा पाहण्याचा संदर्भ फ्रेम स्थापित झाल्यानंतर, जागतिक समन्वयातील वर्णनांचे रूपांतर दृश्य समन्वयांमध्ये करा.
५. सामान्यीकृत को-ऑर्डिनेट्समध्ये दृश्य पोर्ट परिभाषित करा आणि दृश्याचे दृश्य समन्वयक वर्णन सामान्यीकृत समन्वयांमध्ये मॅप करा.
६. व्ह्यूपोर्टच्या बाहेर असलेले चित्राचे सर्व भाग क्लिप करा.

४.२ लाइन क्लिपिंग अल्गोरिदम

ओळ क्लिपिंग(Line Clipping):

संगणक ग्राफिक्समध्ये, लाईन क्लिपिंग ही रूची असलेल्या क्षेत्राबाहेरील रेषा किंवा रेषांचे काही भाग काढून टाकण्याची (क्लिपिंग) प्रक्रिया आहे. संगणक ग्राफिक्समधील क्लिपिंगचा प्राथमिक वापर दृश्य उपखंडाच्या बाहेर असलेल्या वस्तू, रेषा किंवा रेषाखंड काढून टाकणे आहे.

१:कोहेन सदरलँड ओळ क्लिपिंग अल्गोरिदम

श्रेणी किंवा ओळीचा प्रकार(ओळीची दृश्यमानता)

i) प्रकार १: पूर्णपणे आत(पूर्णपणे दृश्यमान)

ii) प्रकार २: पूर्णपणे बाहेर.(अदृश्य)

iii) प्रकार 3:अंशतः आत (अंशतः दृश्यमान)

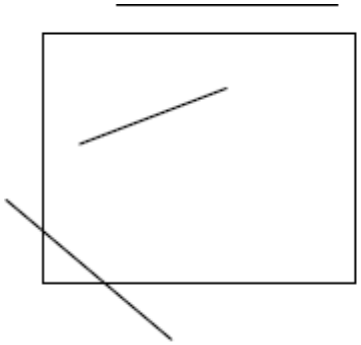


Fig (a) Before clipping

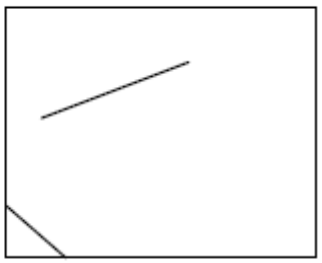


Fig (b) After clipping

क्लिपिंग करण्यापूर्वी **क्लिपिंग नंतर**

आकृती ३ ओळ क्लिपिंग

वरील आकृतीमध्ये तीन ओळी आहेत दिले मध्ये जी एक ओळ आहे ती विंडोव च्या आत आहे त्यामुळे प्रदर्शित आहे, दुसरी ओळ पूर्णपणे बाहेर आहे तर ती काढून टाकली आहे आणि तिसऱ्या ओळ हि क्लीप्पिंग साठी वापरली जाते .कारण तीचा काही भाग विंडोव मध्ये आहे आणि काही भाग विंडोव च्या बाहेर आहे.संगणक ग्राफिक्समध्ये, कोहेन-सुदरलँड अल्गोरिदम हे लाइन क्लिपिंगसाठी वापरले जाणारे अल्गोरिदम आहे. अल्गोरिदम द्विमितीय जागेला 9 क्षेत्रांमध्ये विभाजित करतो आणि नंतर रूचीच्या मध्यवर्ती प्रदेशात (व्ह्यूपोर्ट) दृश्यमान असलेल्या रेषा आणि रेषांचे भाग कार्यक्षमतेने निर्धारित करते.

Line	Code for End point		Logical And Operation	Result
P1, P2	0000	0000	0000	Completely Visible
P3,P4	0001	0001	0001	Completely invisible
P5,P6	0001	0000	0000	Partly Visible
P7,P8	0100	0010	0000	Partly Visible
P9,P10	1000	0010	0000	Partly Visible

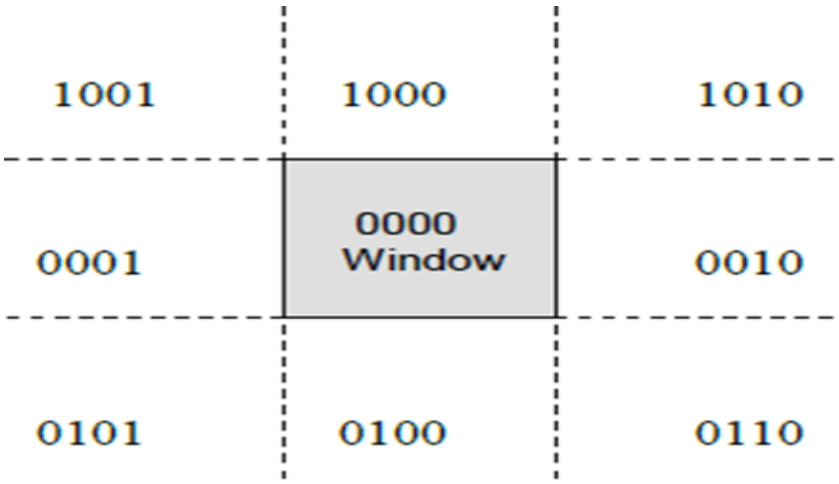


Fig Four bit codes for nine regions **(९ प्रदेश)**

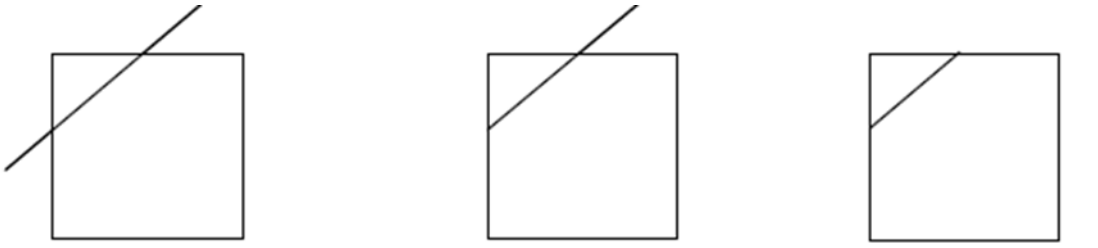


Fig (a) actual line

Fig (b) clipping one part of line

Fig (c) clipping remaining part.

a)वास्तविक ओळ ओळीचा **b) एक भाग कापणे रेषेचा** **c) उरलेला भाग क्लिपिंग**

आकृती ४ ओळ क्लिपिंग

अल्गोरिदम:

1. p1(x1,y1) आणि p2(x2,y2) असे 2 शेवटचे बिंदू वाचा.

2. (wx1,wy1) आणि (wx2,wy2) म्हणून क्लिपिंग विंडोचे 2 कोपरे बिंदू (डाव्या-वर आणि उजव्या-खाली) वाचा.

3. खालील चरणांचा वापर करून 2 एंडपॉइंट्स p1 आणि p2 साठी प्रदेश कोड नियुक्त करा:- 0000 सह कोड आरंभ करा.

x<wx1 असल्यास बिट 1 सेट करा x>wx2 असल्यास बिट 2 सेट करा

3 असल्यास y<wy2 बिट 4 जर y>wy1 सेट करा

4. ओळीची दृश्यमानता तपासा.

a) जर दोन्ही एंडपॉइंट्सचे क्षेत्र कोड शून्य असतील तर रेषा पूर्णपणे दृश्यमान असेल. पायरी 9 वर जा रेषा काढा.

b)जर एंडपॉइंट्सचे क्षेत्र कोड शून्य नसतील आणि त्यांचे तार्किक ANDing देखील शून्य असेल तर रेखा अदृश्य आहे. ओळ टाकून द्या आणि पायरी 9 वर जा.

c)जर ते 4.a आणि 4.b ची पूर्तता करत नसेल तर रेषा अंशतः दृश्यमान आहे.

5. खालीलप्रमाणे क्लिपिंग विंडोचा छेदणारा किनारा निश्चित करा:-

a दोन्ही एंडपॉइंट्सचे क्षेत्र कोड शून्य असल्यास सीमा किनार्यांसह छेदनबिंदू p1′ आणि p2′ शोधा.

b जर कोणत्याही एका टोकाच्या बिंदूचे क्षेत्र कोड शून्य नसतील तर छेदनबिंदू p1′ किंवा p2′ शोधा.

6. छेदनबिंदू विचारात घेऊन रेषाखंड विभाजित करा.

7. रेषेचा कोणताही शेवटचा बिंदू कोणत्याही सीमेच्या बाहेर दिसल्यास रेषा खंड नाकारा.

8. कापलेला रेषाखंड काढा.

9. थांबा.

Program:

```
#include<stdio.h> #include<graphics.h

> void main()

{

int gd=DETECT, gm;

float i,xmax,ymax,xmin,ymin,x11,y11,x22,y22,m; float a[4],b[4],c[4],x1,y1;

clrscr(); initgraph(&gd,&gm,"c:\\turboc3\\bgi"

);

printf("\nEnter the bottom-left coordinate of viewport: "); scanf("%f %f",&xmin,&ymin);

printf("\nEnter the top-right coordinate of viewport: "); scanf("%f %f",&xmax,&ymax); rectangle(xmin,ymin,xmax,ymax);

printf("\nEnter the coordinates of 1st end point of line: "); scanf("%f %f",&x11,&y11);

printf("\nEnter the coordinates of 2nd endpoint of line: "); scanf("%f %f",&x22,&y22);

line(x11,y11,x22,y22); for(i=0;i<4;i++)

{
```

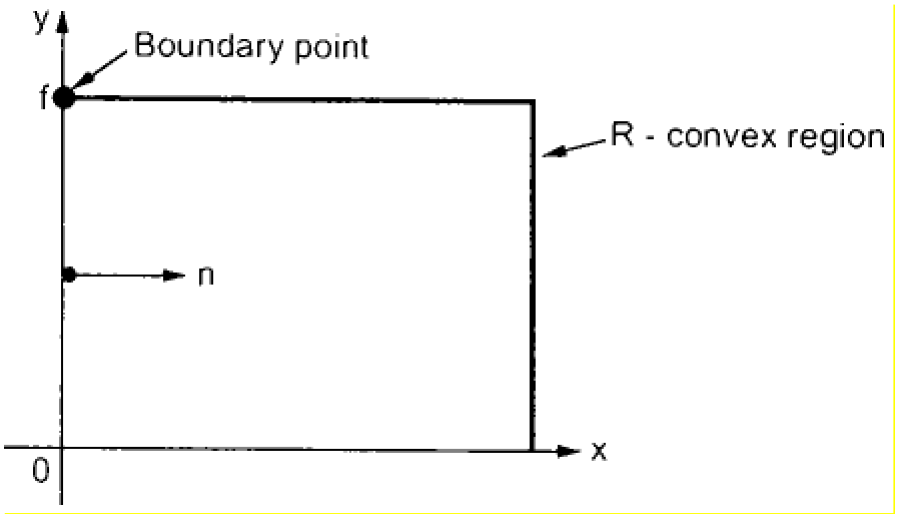
```
a[i]=0;

b[i]=0;

}

m=(y22-y11)/(x22-x11);

if(x11<xmin) a[3]=1; if(x11>xmax) a[2]=1; if(y11<ymin) a[1]=1; if(y11>ymax) a[0]=1;
if(x22<xmin) b[3]=1; if(x22>xmax) b[2]=1; if(y22<ymin) b[1]=1; if(y22>ymax) b[0]=1;
printf("\nRegion code of 1st pt "); for(i=0;i<4;i++)

 {printf("%f",a[i]);} printf("\nRegion code of 2nd pt "); for(i=0;i<4;i++)

{printf("%f",b[i]);} printf("\nAnding :

");

for(i=0;i<4;i++)

{c[i]=a[i]&&b[i];} for(i=0;i<4;i++) printf("%f",c[i]); getch();

if((c[0]==0)&&(c[1]==0)&&(c[2]==0)&&(c[3]==0))

{

if((a[0]==0)&&(a[1]==0)&&(a[2]==0)&&(a[3]==0)&&

(b[0]==0)&&(b[1]==0)&&(b[2]==0)&&(b[3]==0))

{

clrscr(); clearviewport();

printf("\nThe line is totally visible\nand not a clipping candidate");
rectangle(xmin,ymin,xmax,ymax);

line(x11,y11,x22,y22); getch();

}

else

{

clrscr(); clearviewport();
```

```
printf("\nLine is partially visible"); rectangle(xmin,ymin,xmax,ymax); line(x11,y11,x22,y22);

getch(); if((a[0]==0)&&(a[1]==1))

{

x1=x11+(ymin-y11)/m; x11=x1;

y11=ymin;

}

else if((b[0]==0)&&(b[1]==1))

{

x1=x22+(ymin-y22)/m; x22=x1;

y22=ymin;

}

 if((a[0]==1)&&(a[1]==0))

{

x1=x11+(ymax-y11)/m; x11=x1; y11=ymax;

}

else if((b[0]==1)&&(b[1]==0))

{

x1=x22+(ymax-y22)/m; x22=x1; y22=ymax;

} if((a[2]==0)&&(a[3]==1))

{

y1=y11+m*(xmin-x11); y11=y1; x11=xmin;

}

else if((b[2]==0)&&(b[3]==1))

{

y1=y22+m*(xmin- x22); y22=y1;
```

```
x22=xmin;
} if((a[2]==1)&&(a[3]== 0))
{
y1=y11+m*(xmax-x11); y11=y1; x11=xmax;
}
else if((b[2]==1)&&(b[3]==0))
{
y1=y22+m*(xmax-x22); y22=y1;
x22=xmax;
}
clrscr(); clearviewport();
printf("\nAfter clippling:"); rectangle(xmin,ymin,xmax,ymax); line(x11,y11,x22,y22);
getch();
}
}
else
{
 clrscr(); clearviewport();
printf("\nLine is invisible"); rectangle(xmin,ymin,xmax,ymax); getch();
}
closegraph(); getch();
}
```

२: सायरस-बेक अल्गोरिदम

बॉउंद्री पॉईंट कॉन्व्हेक्स region



आकृती ४ सायरस-बेक_ओळ क्लिपिंग

सायरस बेक एक लाइन क्लिपिंग अल्गोरिदम आहे जो बहिर्वक्र बहुभुजांसाठी बनविला जातो. हे कोहेन सदरलँड किंवा निकोल ले निकोलच्या विपरीत, आयताकृती नसलेल्या खिडक्यांसाठी लाइन क्लिपिंगला अनुमती देते. हे कोहेन सदरलँडमध्ये आवश्यक पुनरावृत्ती क्लिपिंग देखील काढून टाकते.सायरस आणि बेक यांनी अनियंत्रित बहिर्वक्र प्रदेशांना क्लिपिंगसाठी अल्गोरिदम विकसित केले. सायरस- बेक अल्गोरिदम बिंदू विन्डोव्याआत, वर किंवा बाहेर आहे की नाही हे निर्धारित करण्यासाठी सामान्य वेक्टर वापरतो.

बहिर्वक्र प्लॅनर बहुभुज R विचारात घ्या. P1 ते P2 रेषेचे पॅरामेट्रिक प्रतिनिधित्व विचारात घ्या.

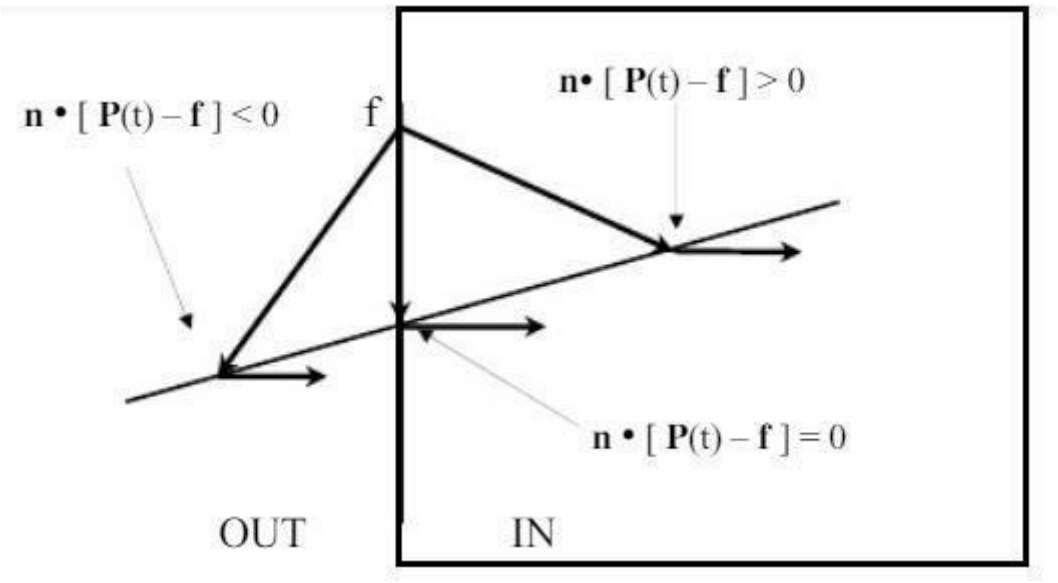
$$P(t) = P_1 + (P_2 - P_1)\,t \qquad 0 \le t \le 1$$

जर 'f' बहिर्वक्र प्रदेश R चा सीमा बिंदू असेल आणि n हा त्याच्या एका सीमारेषेसाठी आतील सामान्य असेल, तर t च्या कोणत्याही विशिष्ट मूल्यासाठी, म्हणजे P1P2 रेषेवरील कोणत्याही विशिष्ट बिंदूसाठी.

$\mathbf{n} \bullet [\,\mathbf{P}(t) - \mathbf{f}\,] < 0$ implies that the vector $\mathbf{P}(t) - \mathbf{f}$ is pointed away from the interior of R,

$\mathbf{n} \bullet [\,\mathbf{P}(t) - \mathbf{f}\,] = 0$ implies that $\mathbf{P}(t) - \mathbf{f}$ is perpendicular to the normal,

$\mathbf{n} \bullet [\,\mathbf{P}(t) - \mathbf{f}\,] > 0$ implies that the vector $\mathbf{P}(t) - \mathbf{f}$ is pointed toward the interior of R.



आकृती ५ सायरस-बेक_ओळ क्लिपिंग

या परिस्थिती एकत्रितपणे दर्शवतात की बहिर्वक्र बहुभुजासाठी, प्रदेशाच्या सीमारेषेला छेदणारी असीम रेषा, दोन बिंदूंवर असे करते. पुढे, हे दोन बिंदू एकाच सीमेच्या काठावर नाहीत. अशा प्रकारे n• [P(t) –f] = 0 ला एकच उपाय आहे. जर बिंदू 'f' सीमारेषेच्या काठावर असेल ज्यासाठी n हा आतील सामान्य आहे, तर P(t) रेषेवरील तो बिंदू (t) जो ही स्थिती पूर्ण करतो तो रेषेचा छेदनबिंदू आणि सीमावर्ती किनार आहे.

<u>अल्गोरिदम:</u>

१. 1.प्रत्येक काठाचे नॉर्मल मोजले जाते.
२. 2.क्लिपिंग लाइनसाठी वेक्टरची गणना केली जाते.
३. 3.प्रति धार एका शिरोबिंदूचा फरक आणि क्लिपिंग लाइनचा एक निवडलेला शेवटचा बिंदू आणि काठाचा सामान्य यांच्यातील बिंदू उत्पादनाची गणना केली जाते (सर्व कडांसाठी).
४. 4.क्लिपिंग लाइनच्या वेक्टर आणि काठाच्या सामान्य (सर्व कडांसाठी) मधील डॉट उत्पादनाची गणना केली जाते.
५. 5.पूर्वीच्या बिंदू उत्पादनास नंतरच्या बिंदू उत्पादनाने भागले जाते आणि -1 ने गुणाकार केला जातो. हे 'टी' आहे.
६. 6.'t' ची मूल्ये त्यांच्या भाजकांचे (नंतरचे बिंदू उत्पादन) निरीक्षण करून प्रवेश करणे किंवा बाहेर पडणे (सर्व किनार्यांमधून) म्हणून वर्गीकृत केले जाते.
७. 7.प्रत्येक गटातून 't' चे एक मूल्य निवडले जाते आणि निर्देशांकांची गणना करण्यासाठी एका रेषेच्या पॅरामेट्रिक फॉर्ममध्ये ठेवले जाते.
८. 8.जर एंटर केलेले 't' मूल्य बाहेर पडणाऱ्या 't' मूल्यापेक्षा मोठे असेल, तर क्लिपिंग लाइन नाकारली जाईल.

3:लियांग- बार्स्की ओळ क्लिपिंग

लिआंग-बार्स्की अल्गोरिदम एक लाइन क्लिपिंग अल्गोरिदम आहे. हे अल्गोरिदम कोहेन-सुदरलँड लाइन क्लिपिंग अल्गोरिदमपेक्षा अधिक कार्यक्षम आहे आणि ते 3-आयामी क्लिपिंगपर्यंत वाढवले जाऊ शकते. हा अल्गोरिदम वेगवान पॅरामेट्रिक लाइन-क्लिपिंग अल्गोरिदम मानला जातो.

• लिआंग-बार्स्की आणि सायरस-बेक यांच्या क्लिपिंग लाइनच्या कल्पना समान आहेत. फरक एवढाच आहे की लिआंग-बार्स्की अल्गोरिदम सरळ आयताकृती क्लिप विंडोसाठी ऑप्टिमाइझ केले गेले आहे.

• लिआंग आणि बार्स्की यांनी एक अल्गोरिदम तयार केला आहे जो फ्लोटिंग-पॉइंट अंकगणित वापरतो परंतु जास्तीत जास्त चार गणनेसह योग्य अंतिम बिंदू शोधतो.

• हा अल्गोरिदम एका रेषेसाठी पॅरामेट्रिक समीकरणे वापरतो आणि ज्या पॅरामीटरसाठी व्ह्यूपोर्टमध्ये आहे त्या पॅरामीटरची श्रेणी शोधण्यासाठी चार असमानता सोडवते.

या क्लिपिंगमध्ये खालील संकल्पना वापरल्या आहेत:

रेषेचे पॅरामेट्रिक समीकरण.

क्लिपिंग विंडोच्या श्रेणीचे वर्णन करणारी असमानता जी रेखा आणि क्लिप विंडोमधील छेदनबिंदू निर्धारित करण्यासाठी वापरली जाते.

रेषेचे पॅरामेट्रिक समीकरण याद्वारे दिले जाऊ शकते,

X = x1 + t(x2-x1)

Y = y1 + t(y2-y1)

जेथे, टी 0 आणि 1 दरम्यान आहे.

नंतर, पॅरामेट्रिक फॉर्ममध्ये पॉइंट-क्लिपिंग अटी लिहा:

xwmin <= x1 + t(x2-x1) <= xwmax

ywmin <= y1 + t(y2-y1) <= ywmax

वरील 4 असमानता याप्रमाणे व्यक्त केल्या जाऊ शकतात,

tpk <= qk

जेथे k = 1, 2, 3, 4 (अनुक्रमे डावीकडे, उजवीकडे, तळाशी आणि वरच्या सीमांना अनुरूप).

p आणि q ची व्याख्या अशी केली आहे,

p1 = -(x2-x1), q1 = x1 - xwmin (डावी सीमा)

p2 = (x2-x1), q2 = xwmax - x1 (उजवी सीमा)

p3 = -(y2-y1), q3 = y1 - ywmin (तळाशी सीमा)

p4 = (y2-y1), q4 = ywmax - y1 (शीर्ष सीमा)

जेव्हा रेषा दृश्य विंडोच्या सीमारेषेला समांतर असते, तेव्हा त्या सीमारेषेसाठी p मूल्य शून्य असते.

जेव्हा pk < 0, टी वाढवणारी रेषा बाहेरून आत जाते (प्रवेश करताना).

जेव्हा pk > 0, ओळ आतून बाहेरून जाते (बाहेर पडते).

जेव्हा pk = 0 आणि qk < 0 तेव्हा रेषा क्षुल्लकपणे अदृश्य असते कारण ती दृश्य विंडोच्या बाहेर असते.

जेव्हा pk = 0 आणि qk > 0 असेल तेव्हा रेषा संबंधित विंडो सीमेच्या आत असते.

<u>अल्गोरिदम -</u>

tmin=0, tmax=1 सेट करा.

t (t(डावीकडे), t(उजवीकडे), t(टॉप), t(तळाशी) च्या मूल्यांची गणना करा.

(i) जर t < tmin त्याकडे दुर्लक्ष करा आणि पुढच्या काठावर जा.

(ii) अन्यथा आतील उत्पादनाचा वापर करून टी मूल्ये प्रविष्ट करणे किंवा बाहेर पडणारी मूल्ये म्हणून वेगळे करा.

(iii) जर t मूल्य प्रविष्ट करत असेल, तर tmin = t सेट करा; टी विद्यमान मूल्य असल्यास, tmax = t सेट करा.

tmin < tmax असल्यास, (x1 + tmin(x2-x1), y1 + tmin(y2-y1)) पासून (x1 + tmax(x2-x1), y1 + tmax(y2-y1)) पर्यंत एक रेषा काढा

ओळ विंडोव वर ओलांडल्यास, (x1 + tmin(x2-x1), y1 + tmin(y2-y1)) आणि (x1 + tmax(x2-x1), y1 + tmax(y2-y1)) हे रेषा आणि काठाचा बिंदू छेदनबिंदू आहेत.

रेषाखंडाचे पॅरामेट्रिक समीकरण Details:

१. X = X1 + U ΔX Y = Y1 + U ΔY
२. कुठे, ΔX = X2 – X1 आणि ΔY = Y2 – Y1
३. 2. या समीकरणांचा वापर करून सायरस आणि बेक यांनी एक अल्गोरिदम विकसित केला जो कोहेन सदरलँड अल्गोरिदमपेक्षा अधिक कार्यक्षम बनवला जातो.
४. नंतर लिआंग आणि बार्स्की यांनी स्वतंत्रपणे आणखी वेगवान पॅरामेट्रिक लाइन क्लिपिंग अल्गोरिदम तयार केले.

५. लिआंग-बार्स्की पध्दतीमध्ये आपण प्रथम पॅरामेट्रिक फॉर्ममध्ये पॉइंट क्लिपिंग कंडिशन करतो: Xmin≤X1+UΔX≤XmaxXmin≤X1+UΔX≤Xmax

Ymin≤Y1+UΔY≤YmaxYmin≤Y1+UΔY≤Ymax

६. या चार असमानतांपैकी प्रत्येक असमानता याप्रमाणे व्यक्त केली जाऊ शकते: k=1,2,3,4 साठी μpk ≤ qk

७. p आणि q हे पॅरामीटर्स खालीलप्रमाणे परिभाषित केले आहेत:

८. p1 = -ΔX आणि q1 = X1 – Xmin (डावी सीमा) p2 = ΔX आणि q2 = Xmax - x1 (उजवी सीमा) P3 = -ΔY आणि q3 = Y1- Ymin (तळाशी सीमा) P4 = ΔY आणि q4 = Ymax - y1 (शीर्ष सीमा)

९. जर एखादी रेषा दृश्य विंडोच्या सीमारेषेला समांतर असेल, तर त्या सीमारेषेसाठी p मूल्य शून्य आहे.

१०. जर रेषा X अक्षाच्या समांतर असेल, उदाहरणार्थ p1 आणि p2 शून्य असणे आवश्यक आहे.

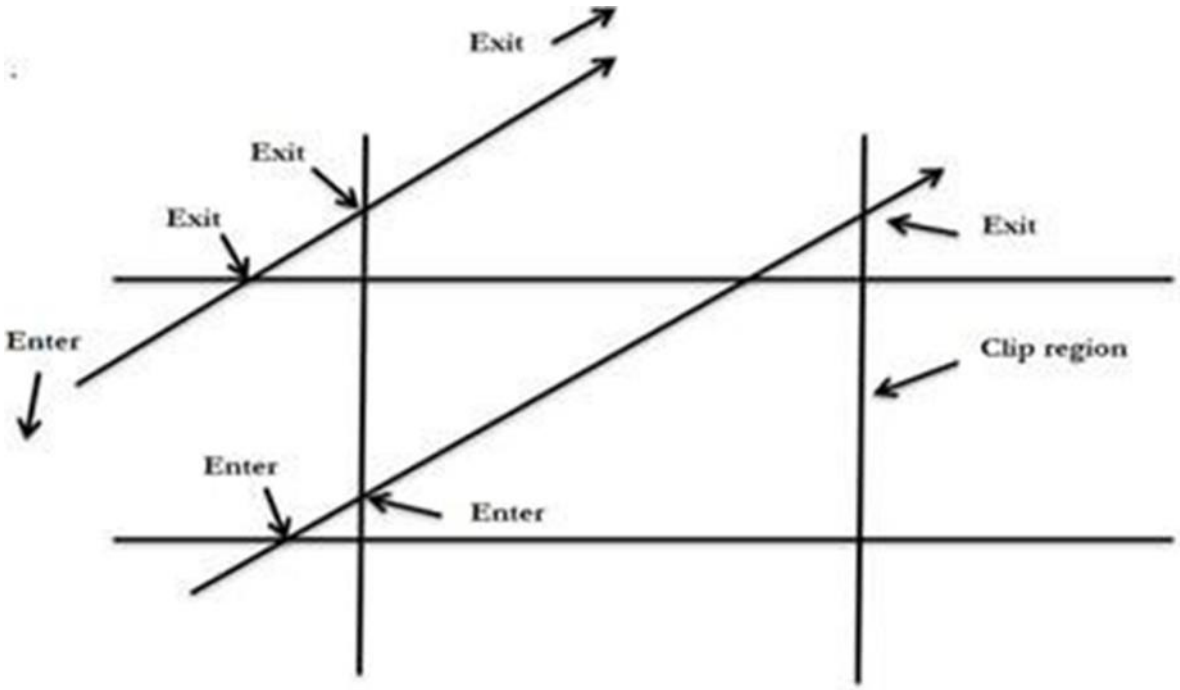
११. pk = 0 दिल्यास, जर qk < 0 असेल तर रेखा क्षुल्लकपणे अदृश्य आहे कारण ती दृश्य विंडोच्या बाहेर आहे.

१२. दिलेले pk = 0, जर qk > 0 असेल तर रेषा संबंधित विंडो सीमेच्या आत आहे.

१३. जेव्हा pk < 0, U वाढलेली रेषा बाहेरून आत जाते म्हणजे प्रवेश करते.

१४. जेव्हा pk > 0, रेखांश आतून बाहेरून म्हणजेच बाहेर पडतात.

१५. जर क्लिप प्रदेशात रेषेचा एक भाग असेल तर, आकृतीमध्ये दर्शविल्याप्रमाणे अनंत रेषेच्या छेदनबिंदूंचा क्रम आत जाणे, प्रवेश करणे, बाहेर येणे आणि बाहेर येणे आवश्यक आहे.



आकृती ६ लियांग- बार्स्की ओळ क्लिपिंग

फायदे

1. इतर अल्गोरिदमपेक्षा अधिक कार्यक्षम कारण सीमांच्या गणनेसह रेखा छेदन कमी केले जाते.

2. रेषेच्या छेदनबिंदूंची गणना एकदाच केली जाते.

Program for Line Clipping Using Liang Barsky Algorithm:

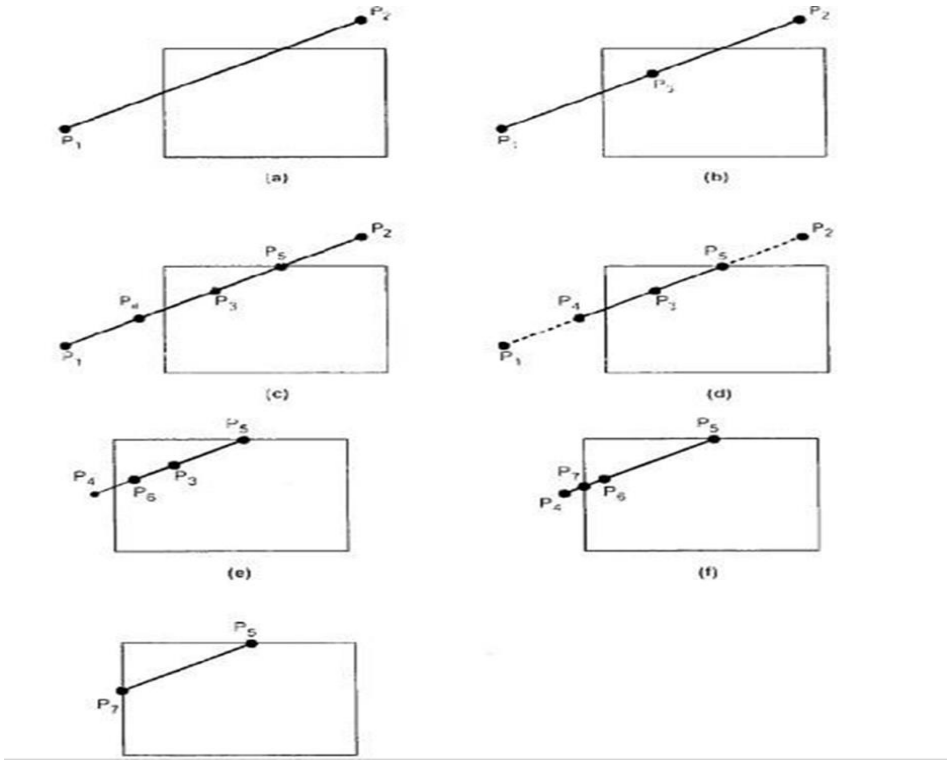
```
#include<stdio.h> #include<conio.h> #include< graphics.h> #include<dos.h>
#define ROUND(a)((int)(a+0.5))
int cliptest(float p,float q,float *u1,float *u2)
{
float r; int retval=1; if(p<0.0)
{
r=q/p; if(r>*u2
)
retval=0; if(r>*u1)
*u1=r;
}
else if(p>0.0)
{
r=q/p; if(r<*u1
)
retval=0
; if(r<*u2)
*u2=r;
}
else if(q<0.0)
retval=0
; return(retval);
}
void clipline(int minx,int miny,int maxx,int maxy,int x1,int y1,int x2,int y2)
{
float u1=0.0,u2=1.0,dx=x2-x1,dy; if(cliptest(-dx,x1- minx,&u1,&u2))
if(cliptest(dx,maxx- x1,&u1,&u2))
{
dy=y2-y1;
```

```
if(cliptest(-dy,y1- miny,&u1,&u2)) if(cliptest(dy,maxy- y1,&u1,&u2))
{
if(u2<1.0)
{
x2=x1+u2*dx; y2=y1+u2*dy;
}
if(u1>0.0)
{
x1+=u1*dx; y1+=u1*dy
;
}
}
}
void main()
{
line(x1,y1,x2,y2);
}
 int gdriver=DETECT,gmode,x1,y1,x2,y2,minx,miny,maxx,maxy;
initgraph(&gdriver,&gmode,"c:\\turboc3\\bgi");
clrscr();
printf("Enter the min & max x values: ");
scanf("%d %d",&minx,&maxx);
printf("Enter the min & max y values: "); scanf("%d %d",&miny,&maxy);
printf("Enter the first endpoint: ");
scanf("%d %d",&x1,&y1);
printf("Enter the second endpoint: ");
scanf("%d %d",&x2,&y2);
clrscr();
printf("Before Clipping"); line(x1,y1,x2,y2);
rectangle(minx,maxy,maxx,miny); getch();
clrscr();
printf("After Clipping"); clipline(minx,miny,maxx,maxy,x1,y1,x2,y2
); rectangle(minx,maxy,maxx,miny); getch();
```

}

4: मध्यबिंदू उपविभाग अल्गोरिदम:

(मिडपॉइंट सबडिव्हिजन अल्गोरिदम)मध्यबिंदू उपविभाग अल्गोरिदम हा सायरस बेक अल्गोरिदमचा विस्तार आहे. हे अल्गोरिदम प्रामुख्याने दृश्य पोर्टमध्ये उपस्थित असलेल्या रेषांच्या दृश्यमान क्षेत्रांची गणना करण्यासाठी वापरले जाते .



आकृती ७ मध्यबिंदू उपविभाग ओळ क्लिपिंग

अल्गोरिदम:

१. ओळ P1 (x1,y1) आणि P2 (x2,y2) चे दोन शेवटचे बिंदू वाचा.
२. विंडोचे कोपरे वाचा (Wx1, Wy1) आणि (Wx2, Wy2).
३. P1 आणि P2 साठी प्रदेश कोड नियुक्त करा. प्रदेश कोड हा 4 अंकी बिट कोड असतो जो नऊ प्रदेशांपैकी एक रेषेचा शेवटचा बिंदू दर्शवतो.
४. बिट्स 0000 सह कोड आरंभ करा.
५. जर बिट 1 सेट करा (x<Wx1) जर बिट 2 सेट करा (x>Wx2) जर बिट 3 सेट करा (y<Wy2) जर बिट 4 सेट करा (y>Wy1).
६. लाइन P1P2 ची दृश्यमानता तपासा.
    a. P1 आणि P2 दोन्ही टोकांचे क्षेत्र कोड 0 = > रेषा पूर्णपणे दृश्यमान असल्यास. रेषा काढा आणि पायरी 3 वर जा.
    b. P1 आणि P2 दोन्ही टोकांचे क्षेत्र कोड 0 नसल्यास आणि त्यांचे तार्किक ANDing शून्य देखील नाही => रेखा पूर्णपणे अदृश्य आहे. ओळ नकार द्या आणि पायरी 3 वर जा.

c. जर a) आणि b) दोन्ही समाधानी नसतील तर रेषा अंशतः दृश्यमान आहे.

७. अर्धवट दृश्यमान रेषेचे भाग समान भागांमध्ये विभाजित करा आणि दोन्ही उपविभाजित रेषेसाठी 3 ते 5 चरणांची पुनरावृत्ती करा जोपर्यंत तुम्हाला पूर्णपणे दृश्यमान आणि पूर्णपणे अदृश्य रेषा विभाग मिळत नाहीत.

८. थांबा.

Program:

```
//MIDPOINT SUBDIVISION LINE CLIPPING

#include<stdio.h> #include<conio.h> #include<stdlib.h> #include<dos.h>
#include<math.h> #include<graphics.h> typedef struct coordinate

{

int x,y;

char code[4];

}PT;

void drawwindow();

void drawline (PT p1,PT p2,int cl); PT setcode(PT p);

int visibility (PT p1,PT p2); PT resetendpt (PT p1,PT p2); main()

{

int gd=DETECT, gm,v;

PT p1,p2,ptemp;

initgraph(&gd,&gm,"C :\\turboc3\\BGI ");

cleardevice();

printf("\n\n\t\tENTER END-POINT 1 (x,y): "); scanf("%d%d",&p1.x,&p1.y);
printf("\n\n\t\tENTER END-POINT 2 (x,y): "); scanf("%d%d",&p2.x,&p2.y); cleardevice();

drawwindow(); getch(); drawline(p1,p2,15

); getch();

 cleardevice(); drawwindow(
```

```
);
midsub(p1,p2
); getch(); closegraph(); return(0);
}
midsub(PT p1,PT p2)
{
PT mid; int v;
p1=setcode(p1); p2=setcode(p2); v=visibility(p1,p2
); switch(v)
{
case 0: /* Line conpletely visible */ drawline(p1,p2,15);
break;
case 1: /* Line completely invisible */ break;
case 2: /* line partly visible */ mid.x = p1.x + (p2.x- p1.x)/2;
mid.y = p1.y + (p2.y-p1.y)/2;
midsub(p1,mid)
; mid.x = mid.x+1; mid.y = mid.y+1; midsub(mid,p2)
; break;
}
}
void drawwindow()
{
setcolor(RED); line(150,100,450,100
); line(450,100,450,400
); line(450,400,150,400
```

```
 ); line(150,400,150,100
);
}
void drawline (PT p1,PT p2,int cl)
{
setcolor(cl); line(p1.x,p1.y,p2.x,p2.y)
;
}
PT setcode(PT p)
{
PT ptemp; if(p.y<=100
)
ptemp.code[0]='1'; /* TOP */ else
ptemp.code[0]='0'; if(p.y>=400) ptemp.code[1]='1'; /* BOTTOM
*/ else ptemp.code[1]='0'; if (p.x>=450)
ptemp.code[2]='1'; /* RIGHT
*/ else ptemp.code[2]='0';
if (p.x<=150) /* LEFT */ ptemp.code[3]='1';
else
ptemp.code[3]='0'; ptemp.x=p.x; ptemp.y=p.y; return(ptemp);
}
int visibility (PT p1,PT p2)
{
int i,flag=0; for(i=0;i<4;i++
)
```

```
{
if((p1.code[i]!='0')||(p2.code[i]!='0')) flag=1;
}
if(flag==0)
 return(0);
for(i=0;i<4;i++)
{
if((p1.code[i]==p2.code[i])&&(p1.code[i]=='1')) flag=0;
}
if(flag==0)
return(1); return(2);
}
```

४.३ पॉलीगोन क्लिपिंग अल्गोरिदम (Polygons Cliping):

बहुभुज क्लिपिंग:

अनेक सुप्रसिद्ध बहुभुज क्लिपिंग अल्गोरिदम आहेत, प्रत्येकाची ताकद आणि कमकुवतपणा आहेत. सर्वात जुने (1974 पासून) सदरलँड-हॉजमन अल्गोरिदम असे म्हणतात. त्याच्या मूळ स्वरूपात, ते तुलनेने सोपे आहे. हे दोन महत्त्वाच्या प्रकरणांमध्ये देखील खूप कार्यक्षम आहे, एक म्हणजे जेव्हा बहुभुज पूर्णपणे सीमांच्या आत असतो आणि दुसरा जेव्हा तो पूर्णपणे बाहेर असतो.

सुदरलँड - हॉजमन (पॉलिगॉन)बहुभुज क्लिपिंग:

सुदरलँड हॉजमन पॉलिगॉन क्लिपिंग अल्गोरिदम बहुभुज क्लिपिंगसाठी वापरला जातो. या अल्गोरिदममध्ये, बहुभुजाचे (पोलिगोन)सर्व शिरोबिंदू क्लिपिंग विंडोच्या प्रत्येक काठावर क्लिप केले जातात. बहुभुजाचे नवीन शिरोबिंदू मिळविण्यासाठी बहुभुज विंडोच्या डाव्या काठावर प्रथम बहुभुज क्लिप केला जातो.बहुभुजाच्या प्रत्येक काठाची क्लिप आयताच्या प्रत्येक काठावर चाचणी करणे आवश्यक आहे. हे प्रत्येक bondaryच्या कोपऱ्यावर किंवा काठावर बहुभुजाच्या सीमेवर प्रक्रिया करून केले जाते. सर्व प्रथम संपूर्ण बहुभुज एका काठावर कापला जातो, नंतर परिणामी बहुभुज विचारात घेतला जातो, नंतर बहुभुज दुसऱ्या काठाच्या विरुद्ध मानला जातो, त्याचप्रमाणे सर्व चारही कडांसाठी.

प्रक्रिया करताना चार संभाव्य परिस्थिती(Four possible situations while processing)

१.जर पहिला शिरोबिंदू खिडकीच्या बाहेर असेल, तर दुसरा शिरोबिंदू खिडकीच्या आत असेल. नंतर आउटपुट सूचीमध्ये दुसरा शिरोबिंदू जोडला जातो. विंडो सीमा आणि बहुभुज बाजू (धार) च्या छेदनबिंदूचा बिंदू देखील आउटपुट लाइनमध्ये जोडला जातो.

२.दोन्ही शिरोबिंदू खिडकीच्या सीमेच्या आत असल्यास. नंतर आउटपुट सूचीमध्ये फक्त दुसरा शिरोबिंदू जोडला जातो.

३.जर पहिला शिरोबिंदू खिडकीच्या आत असेल आणि दुसरा खिडकी बाहेरील असेल. विंडोला छेदणारी किनार आउटपुट सूचीमध्ये जोडली जाते.

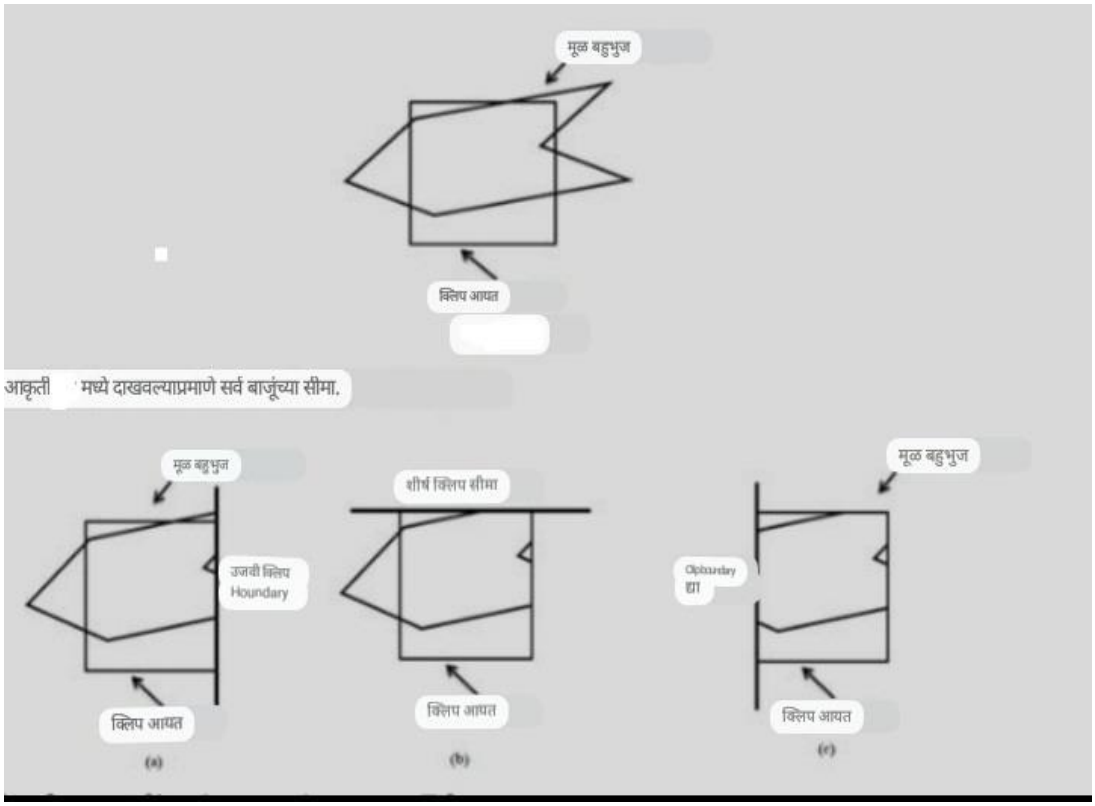
४.जर दोन्ही शिरोबिंदू बाहेरील विंडो असतील, तर आउटपुट सूचीमध्ये काहीही जोडले जात नाही.

आकृती ८ आयत क्लीप्पिंग उदाहरण

क्लीप्पिंग अगोदर **क्लीप्पिंग नंतर**

सुदरलँड -हॉजमन पॉलिगॉन क्लिपिंग अल्गोरिदम:-

प्रत्येक boundary च्या काठावर त्याच्या सीमारेषेवर संपूर्णपणे प्रक्रिया करून बहुभुज क्लिप केला जाऊ शकतो. हे सर्व बहुभुज शिरोबिंदूंवर प्रत्येक क्लिप आयताच्या सीमेवर प्रक्रिया करून प्राप्त केले जाते. बहुभुज शिरोबिंदूंच्या मूळ संचापासून सुरुवात करून, शिरोबिंदूंचा एक नवीन क्रम तयार करण्यासाठी आम्ही प्रथम बहुभुज डाव्या आयताच्या सीमेवर क्लिप करू शकतो. आकृती मध्ये दर्शविल्याप्रमाणे, शिरोबिंदूंचा नवीन संच नंतर क्रमाने उजव्या सीमा क्लिपर, वरच्या सीमा क्लिपर आणि तळाच्या सीमा क्लिपरकडे जाऊ शकतो. प्रत्येक पायरीवर बहुभुज शिरोबिंदूंचा एक नवीन संच तयार केला जातो आणि पुढील विंडो सीमा क्लिपरवर पास केला जातो. सदरलँड - हॉजमन अल्गोरिदममध्ये ही मूलभूत कल्पना वापरली जाते.



आकृती ९ आयत क्लीप्पिंग उदाहरण

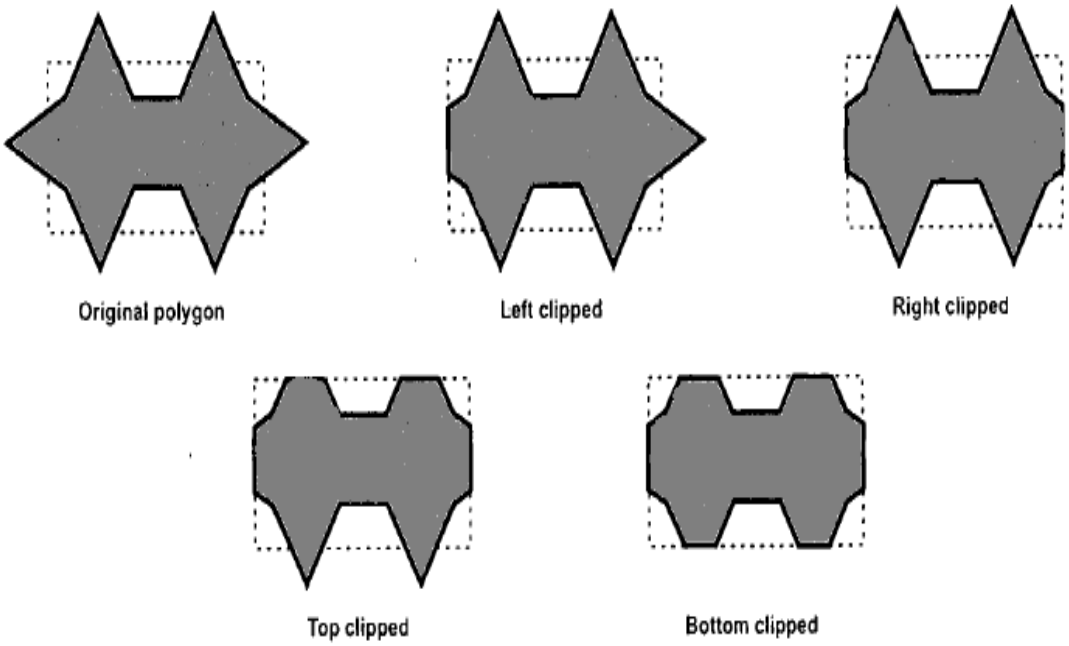


Fig. (l) Clipping a polygon against successive window boundaries

आकृती १० विन्डोव्स च्या सर्व सीमेवर बहुभुजाची क्लिपिंग प्रोसेस

अल्गोरिदमचे आउटपुट बहुभुज शिरोबिंदूंची सूची आहे जे सर्व क्लिपिंग प्लेनच्या दृश्यमान बाजूला आहेत. बहुभुजाच्या अशा प्रत्येक काठाची वैयक्तिकरित्या क्लिपिंग प्लेनशी तुलना केली जाते. क्लिपिंग सीमा किंवा समतल भोवती बहुभुजाच्या प्रत्येक काठाच्या दोन शिरोबिंदूंवर प्रक्रिया करून हे साध्य केले जाते. याचा परिणाम काठ आणि क्लिपिंग सीमा किंवा प्लेन यांच्यातील चार संभाव्य संबंधांमध्ये होतो.जर काठाचा पहिला शिरोबिंदू विंडो सीमेच्या बाहेर असेल आणि काठाचा दुसरा शिरोबिंदू आत असेल तर खिडकीच्या सीमारेषेसह बहुभुज काठाचा छेदनबिंदू आणि दुसरा शिरोबिंदू आउटपुट शिरोबिंदू सूचीमध्ये जोडला जाईल .

एका काठावर बहुभुज क्लिपिंगची चार प्रकरणे: (Four Cases of polygon clipping against one Edge)क्लिपची सीमा दृश्यमान आणि अदृश्य प्रदेश निर्धारित करते. शिरोबिंदूच्या कडा चार प्रकारांपैकी एक असू शकतात:

केस १: जर पहिला शिरोबिंदू खिडकीच्या सीमेच्या बाहेर असेल आणि दुसरा शिरोबिंदू खिडकीच्या आत असेल, तर खिडकीच्या सीमारेषेसह छेदनबिंदू आणि खिडकीच्या आत असलेला शिरोबिंदू आउटपुट शिरोबिंदू सूचीमध्ये संग्रहित केला जाईल.

केस २: जर बहुभुजाचे दोन्ही, पहिले आणि दुसरे शिरोबिंदू खिडकीच्या आत पडलेले असतील, आणि नंतर आपल्याला दुसरा शिरोबिंदू फक्त आउटपुट शिरोबिंदू सूचीमध्ये संग्रहित करावा लागेल.

केस ३: जर पहिला शिरोबिंदू विंडोच्या आत असेल आणि दुसरा शिरोबिंदू खिडकीच्या बाहेर असेल तर आपल्याला आउटपुट शिरोबिंदू सूचीमध्ये विंडोसह बहुभुजाच्या त्या काठाचा फक्त छेदनबिंदू ठेवावा लागेल.

केस ४: जर बहुभुजाचा पहिला आणि दुसरा शिरोबिंदू खिडकीच्या बाहेर पडलेला असेल तर आउटपुट शिरोबिंदू सूचीमध्ये कोणताही शिरोबिंदू संग्रहित केला जात नाही.

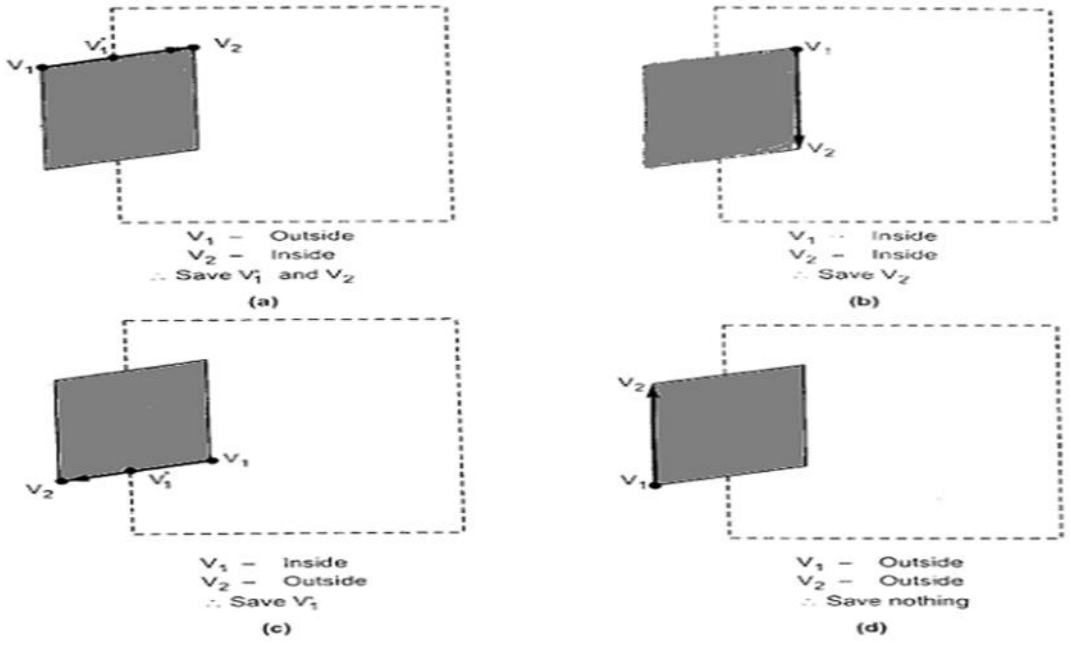
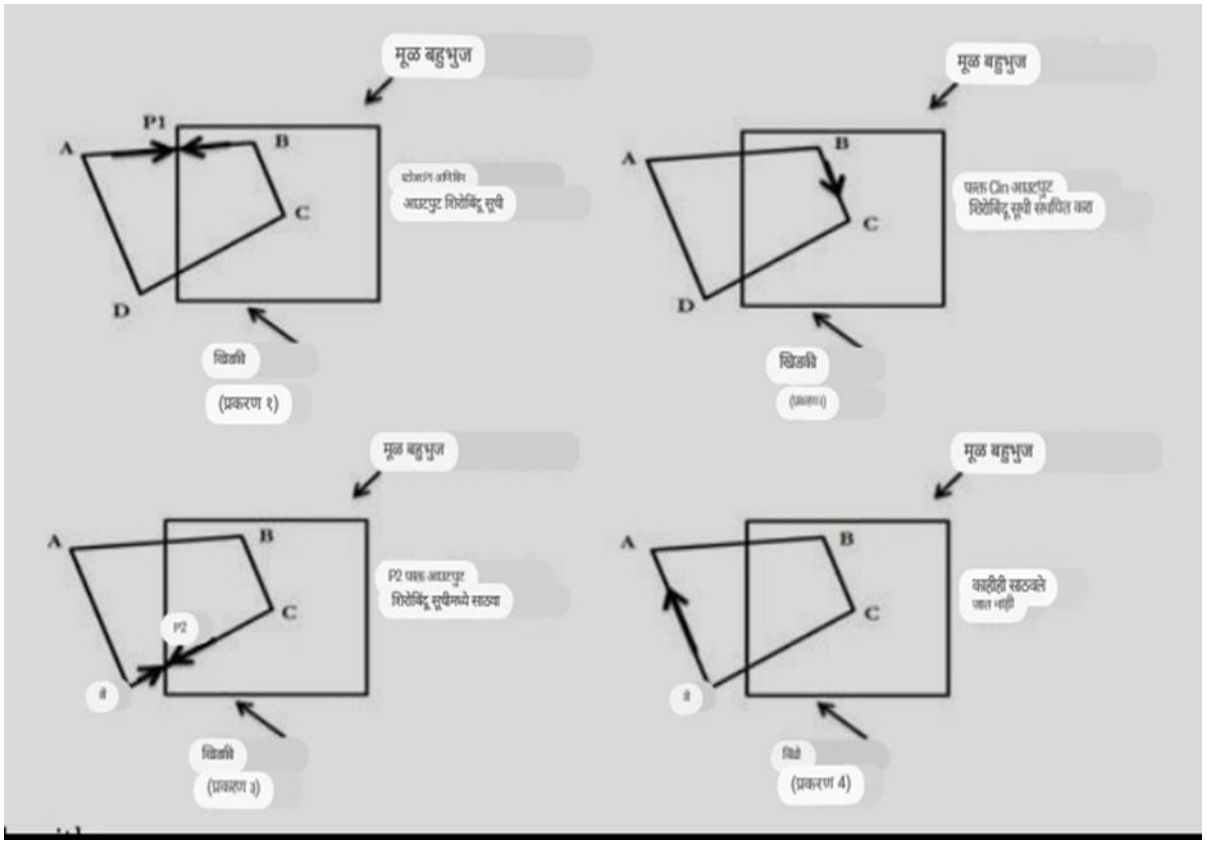


Fig. (m) Processing of edges of the polygon against the left window boundary

आकृती ११ विन्डोव्स च्या सर्व (लेफ्ट,राईट,टोप,बोटम)सीमेवर बहुभुजाची क्लिपिंग प्रोसेस



आकृती १२ विन्डोव्स च्या सर्व (लेफ्ट,राईट,टोप,बोटम)सीमेवर बहुभुजाची क्लिपिंग प्रोसेस

Algorithm:

१. बहुभुजाच्या सर्व शिरोबिंदूंचे निर्देशांक वाचा.
२. क्लिपिंग विंडोचे निर्देशांक वाचा.
३. विंडोव च्या काठावर डावीकडे विचार करा.
४. बहुभुजाच्या प्रत्येक काठाच्या शिरोबिंदूंची वैयक्तिकरित्या क्लिपिंग प्लेनशी तुलना करा
५. परिणामी एज आणि क्लिपिंग सीमा यांच्यातील चार संभाव्य संबंधांनुसार शिरोबिंदूंच्या नवीन सूचीमध्ये परिणामी छेदनबिंदू आणि शिरोबिंदू जतन करा.
६. क्लिपिंग विंडोच्या उरलेल्या कडांसाठी चरण 4 आणि 5 ची पुनरावृत्ती करा. प्रत्येक वेळी क्लिपिंग विंडोच्या पुढील काठावर प्रक्रिया करण्यासाठी शिरोबिंदूंची परिणामी सूची क्रमाने पास केली जाते.
७. थांबा.

Program:

```
// Sutherland Hodgeman Polygon Clipping
#include<stdio.h>
#include<conio.h>
 #include<graphics.h>
```

```
#include<math.h>
void clip(float,float,float); int i,j=0,n;
int rx1,rx2,ry1,ry2; float x1[8],y1[8]; void main()
{
int gd=DETECT,gm; int i,n;
float x[8],y[8],m;
clrscr(); initgraph(&gd,&gm,"c:\\turboc3\\bgi"); printf("coordinates for rectangle : "); scanf("%d%d%d%d",&rx1,&ry1,&rx2,&ry2); printf("no. of sides for polygon : "); scanf("%d",&n);
printf("coordinates : "); for(i=0;i<n;i++)
{
scanf("%f%f",&x[i],&y[i]);
}
cleardevice(); outtextxy(10,10,"Before clipping"); outtextxy(10,470,"Press any key.  ");
rectangle(rx1,ry1,rx2,ry2); for(i=0;i<n-1;i++) line(x[i],y[i],x[i+1],y[i+1]);
line(x[i],y[i],x[0],y[0]); getch();
cleardevice(); for(i=0;i<n-1;i++)
{
m=(y[i+1]-y[i])/(x[i+1]-x[i]); clip(x[i],y[i],m);
clip(x[i+1],y[i+1],m);
}
m=(y[i]-y[0])/(x[i]-x[0]); clip(x[i],y[i],m);
clip(x[0],y[0],m); outtextxy(10,10,"After clipping"); outtextxy(10,470,"Press any key. ");
rectangle(rx1,ry1,rx2,ry2); for(i=0;i<j-1;i++) line(x1[i],y1[i],x1[i+1],y1[i+1]); getch();
}

void clip(float e,float f,float m)
{
while(e<rx1||e>rx2||f<ry1||f>ry2)
{
if(e<rx1)
{
f+=m*(rx1-e); e=rx1;
```

```
}
else if(e>rx2)
{
f+=m*(rx2-e);
 e=rx2;
}
if(f<ry1)
{
e+=(ry1-f)/m; f=ry1;
}
else if(f>ry2)
{
e+=(ry2-f)/m; f=ry2;
}
}
x1[j]=e;
y1[j]=f; j++;
}
```

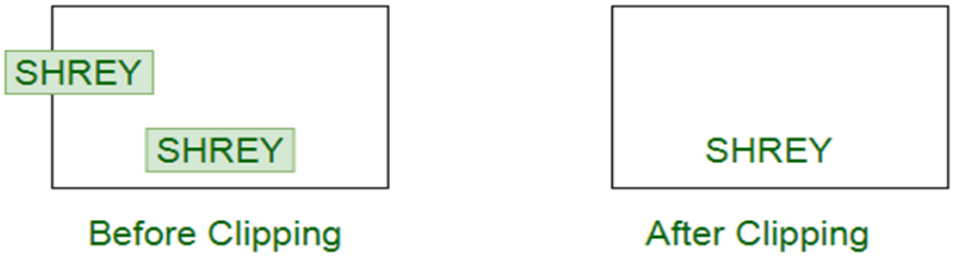
४.४ मजकूर क्लिपिंग (Text Clipping).

मजकूर क्लिपिंग ही स्ट्रिंग क्लिप करण्याची प्रक्रिया आहे. या प्रक्रियेत, अनुप्रयोगाच्या आवश्यकतेनुसार आम्ही संपूर्ण वर्ण किंवा त्यातील काही भाग क्लिप करतो.

मजकूर क्लिपिंग पद्धती:

१.सर्व किंवा कोणतीही स्ट्रिंग क्लिपिंग पद्धत -

या पद्धतीत, जर संपूर्ण स्ट्रिंग क्लिप विंडोच्या आत असेल तर आम्ही त्याचा विचार करतो. अन्यथा, स्ट्रिंग पूर्णपणे काढून टाकली जाईल. मजकूर नमुना एका बाउंडिंग आयत अंतर्गत मानला जातो. नंतर आयताच्या सीमा स्थानांची खिडकीच्या सीमांशी तुलना केली जाते. स्ट्रिंग आणि विंडो यांच्यामध्ये आच्छादन असल्यास स्ट्रिंग नाकारली जाते. ही पद्धत सर्वात वेगवान मजकूर क्लिपिंग तयार करते.

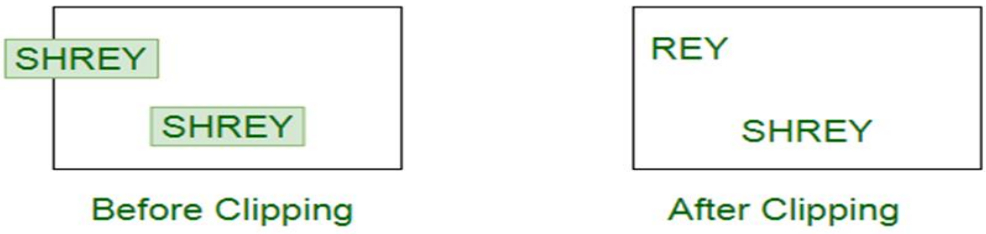


क्लिपिंग करण्यापूर्वी **क्लिपिंग नंतर**

आकृती १३

२.सर्व किंवा काहीही नाही वर्ण क्लिपिंग पद्धत -

या पद्धतीत, आम्ही क्लिप विंडोच्या आत असलेल्या स्ट्रिंगचे अक्षरे ठेवतो आणि क्लिप विंडोच्या बाहेर असलेले सर्व वर्ण काढून टाकतो. वैयक्तिक वर्णांची सीमा मर्यादा विंडोशी तुलना केली जाते. क्लिप विंडोसह वर्ण आच्छादित झाल्यास, आम्ही वर्ण काढून टाकतो.



क्लिपिंग करण्यापूर्वी **क्लिपिंग नंतर**

आकृती १४

३.मजकूर क्लिपिंग पद्धत -

या पद्धतीत, आम्ही क्लिप विंडोच्या आत असलेल्या स्ट्रिंगचे वर्ण ठेवतो आणि क्लिप विंडोच्या बाहेर असलेले सर्व वर्ण काढून टाकतो. जर एखादे अक्षर खिडकीच्या सीमारेषेला ओव्हरलॅप करत असेल तर खिडकीच्या आत असलेला वर्णाचा भाग आम्ही ठेवतो आणि क्लिप विंडोच्या बाहेर असलेला भाग टाकून देतो.

क्लिपिंग करण्यापूर्वी **क्लिपिंग नंतर**

आकृती १५

घटक ५ (Unit 5)

वक्ररेषेचा परिचय

(Introduction to Curve)

(गुण १२)

विषय निष्पत्ती (Course Outcome): अल्गोरिदम वापरून वक्ररेषा तयार करण्यासाठी प्रोग्राम लागू करा.

घटक निष्पत्ती (Unit Outcome):

१. दिलेल्या वक्ररेषा निर्मिती पद्धतीचे वर्णन करा.

२. दिलेल्या वक्ररेषा अल्गोरिदमचा वापर करून वक्ररेषा काढा.

३. दिलेल्या वक्ररेषा स्थितीचे गुणधर्म सांगा .

४. दिलेल्या अल्गोरिदमद्वारे वर्तुळाकार रेषेच्या भागाची (Arc) निर्मिती करा.

५. ५. १ वक्ररेषेची निर्मिती (Curve Generation)

संगनिकय ग्राफिक्स मध्ये बऱ्याच वेळा आपल्याला वेगवेगळ्या प्रकारच्या गोष्टी स्क्रीनवर (screen) काढाव्या लागतात. गोष्टी नेहमी सपाट नसतात आणि एखादी गोष्ट काढण्यासाठी आपल्याला अनेक वेळा वक्ररेषा काढावी लागते. वक्ररेषा हा बिंदूंचा अमर्याद मोठा संच आहे. प्रत्येक बिंदूला शेवटचे बिंदू वगळता दोन शेजारी असतात.

वक्र रेषा काढण्यासाठी आपण दोन पध्दती वापरू शकतो. यामध्ये पहिली पध्दत अशी आहे की ,अल्गोरिदमचा वापर करून वक्ररेषा निर्मिती करणे. जसे DDA अल्गोरिदम , यामध्ये तंतोतंत वक्ररेषा निर्मित केली जाते. दुसऱ्या पध्दती मध्ये वक्र रेषा निर्मित करण्यासाठी अंदाजे अनेक लहान सरळ रेषेच्या रेशाखंडचा (segment) वापर केला जातो आणि यासाठी इंटरपोलशन (Interpolation) तंत्राचा उपयोग केला जातो.

५.१.१ वर्तुळाकार रेषेच्या भागाची (Arc) निर्मिती करण्यासाठी DDA अल्गोरिदम

(Circular Arc Generation algorithm using DDA Algorithm)

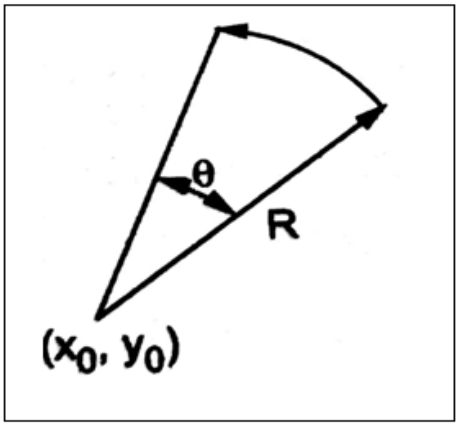
DDA हे Digital Differential Analyzer चे संक्षिप्त रूप आहे. DDA मध्ये गणितातील डीजिटल डीफ्रन्शिअल इकवेषण ऑफ कर्व (Differential Equation of curve) चा उपयोग केला जातो.

चला पाहू या DDA अल्गोरिदम द्वारे वर्तुळाकार रेषेच्या भागाची (Arc) निर्मिती. Angle Parameters नुसार वर्तुळाकार रेषेच्या भागाचे समीकरण खालील प्रमाणे.

$$X = R\cos\theta + x_0$$

$$y = R\sin\theta + y_0 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(५.१)$$

येथे (x_0, y_0) वक्रता केंद्र (center of curvature) आणि R वर्तुळाकार रेषेच्या भागाची त्रिज्या (radius of arc) आकृती क्र . ५. १ पहा.



आकृती क्र . ५. १ वर्तुळाकार रेषेच्या भागाची (Arc) निर्मिती

समीकरण ५.१ ला Differentiate केल्या नंतर खालील समीकरणे मिळतात .

$dx = -R \sin\theta \; d\theta$

$dy = R \cos\theta \; d\theta$ ……(५.२)

समीकरण ५.१ वरून आपण $R \cos\theta$ आणि $R \sin\theta$ खालील प्रमाणे सोडवू शकतो.

$X = R \cos\theta + x_0$

$R \cos\theta = x - + x_0$ आणि $R \sin\theta = y - y_0$ .....(५.३)

समीकरण ५.३ मधून $R \cos\theta$ आणि $R \sin\theta$ मूल्य (values) टाकल्या नंतर आपल्याला खालील समीकरणे मिळतात

$dx = - (y - y_0) \; d\theta$

$dy = (x - x_0) \; d\theta$ .....(५.४)

dx चे मूल्य (value) x मधील वाढ आणि dy चे मूल्य (value) y मधील वाढ (increment) सूचित करतात. arc वर पुढील (next) बिंदू (point) मिळविण्यासाठी ही मूल्ये (Values) वर्तमान (Current) बिंदूमध्ये जोडली जातात.

त्यामुळे आपण लिहू शकतो,

$x_2 = x_1 + dx = x_1 - (y_1 - y_0) \; d\theta$

$y_2 = y_1 + dy = y_1 + (x_2 - x_0) \; d\theta$ .....(५.५)

समीकरण ५. ४ हे arc निर्मिती अल्गोरिदमसाठी आधार आहे.. तसेच अर्क (arc) निर्मितीमध्ये पुढच्या (next) पॉइंट म्हणजेच function of dθ आहे. . गुळगुळीत (smooth) arc निर्मित करण्यासाठी शेजारचे बिंदू एकमेकांच्या जवळ

असले पाहिजेत. हे साध्य करण्यासाठी dθ चे मूल्य (value) लहान असावी आणि त्यामध्ये अंतर असू नये . खालील समीकरणावरून dθ (डी थिटा) ची value ठरवता येते .

$$d\theta = \text{Min}\ (0.01, 1/(3.2\ X\ (\ (x- x_0\ )+ (y- y_0\ )\ )\ )$$

Algorithm

1. Read the center of a curvature, say(x_0 , y_0)
2. Read tha Arc angle say, θ
3. Read the starting point of the arc say(x,y)
4. Calculate dθ

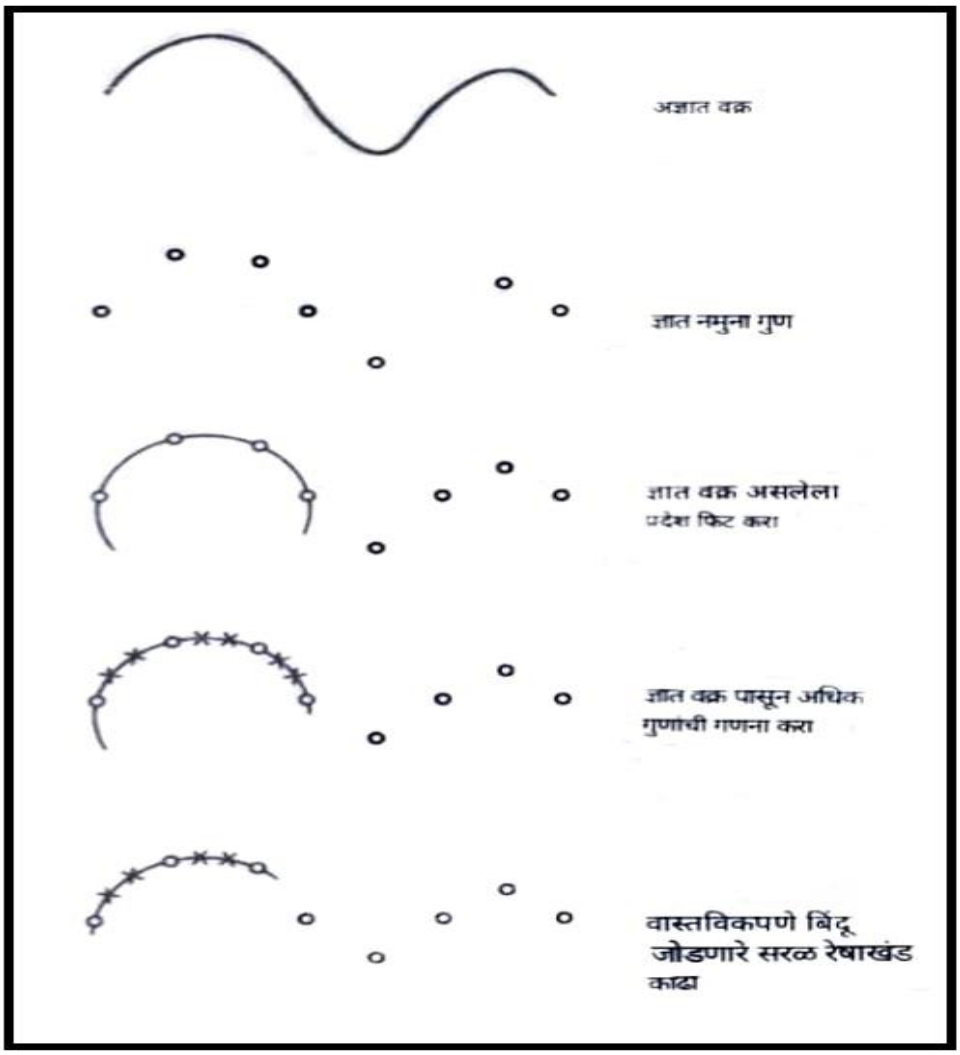
   $d\theta = \text{Min}\ (0.01, 1/(3.2\ X\ (\ (x- x_0\ )+ (y- y_0\ )\ )\ )$

5. Initialize Angel=0
6. While(Angle<0)
   do
   {
   Plot (x,y)
   x =x-(y- y_0) X dθ
   y= y+ (x- x_0) X dθ
   Angle=Angle + d }
7. Stop

५.१.२ इंटरपोलेशन (Interpolation)

वरच्या मुद्द्या मध्ये आपण पहिले की ,DDA अल्गोरिदम वापरून खरी म्हणजेच अचूक वक्ररेषा(Curve) निर्मिती करता येते पण या पध्दती च्या काही मर्यादा आहेत. सराव मध्ये आपल्याला काही जटिल(Complex) वक्ररेषांना सामोरे जावे लागेते ज्यासाठी कोणतेही गणितीय (mathematical) कार्य(function) उपलब्ध नाही. असा प्रकारची वक्ररेषा अंदाजे पद्धती(Approximation Method) वापरून काढता येते.

जर आपल्याकडे नमुना(sample) बिंदूंचा संच आवश्यक वक्र वर स्थित असेल, तर आपण नमुना बिंदूंच्या जवळून जाणार्या ज्ञात(known) वक्रांच्या तुकड्यांमध्ये भाग वक्र भरून आवश्यक वक्र काढू शकतो. नमुना बिंदूंमधील अंतर ज्ञात अंदाजे वक्र बाजूने बिंदूंचे समन्वय शोधून आणि ते बिंदू रेषाखंडांशी जोडून भरले जाऊ शकते. आकृती क्र. ५.२ मध्ये Interpolation ची प्रक्रिया दाखवली आहे



आकृती क्र.५.२ Interpolation ची प्रक्रिया

५.२ वर्करेषेचे प्रकार (Types of Curve)

साधारपणे वर्करेषेचे खालिल प्रकार आहे.

१. हिलबर्ट वक्ररेषा(Hilbert's Curve)

२. कोच वक्ररेषा(Koch Curve)

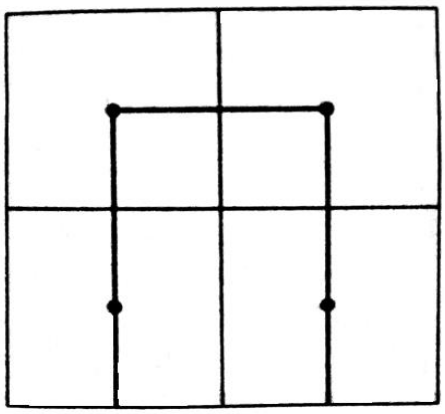
३. बी-Spline वक्ररेषा(B-spline Curve)

४. बेझियर वक्ररेषा(Bezier Curve)

५.२.१ हिलबर्ट वक्ररेषा(Hilbert's Curve)

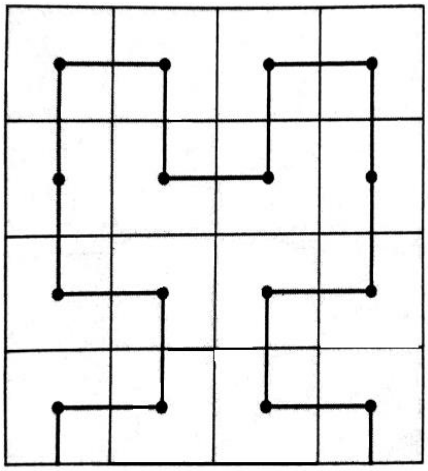
हिलबर्ट वक्ररेषा (हिल्बर्ट स्पेस-फिलिंग वक्र म्हणूनही ओळखले जाते) एक सतत भग्न स्पेस-फिलिंग वक्र आहे ज्याचे वर्णन जर्मन गणितज्ञ डेव्हिड हिल्बर्ट यांनी १८९१ मध्ये केले.

हिलबर्ट वक्ररेषा काढण्यासाठी खालील प्रमाणे क्रमिक अंदाज **(**Successive Approximation) वापरता येतात . जर चौकोनाचे चार चतुर्थांशामध्ये (Four Quadrants) विभाजन केले तर आपल्याला प्रथम अंदाजाने **(**First Approximation) प्रत्येक चतुर्थांशाचा मध्यबिंदू एकमेकांशी जोडून हिलबर्ट वक्ररेषा आकृती क्र.५.३ मध्ये दिल्याप्रमाणे काढता येते.



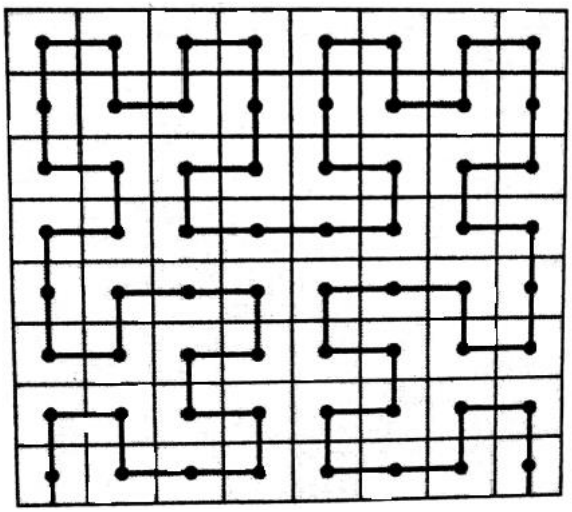
आकृती क्र .५.३ प्रथम अंदाजानुसार (First Approximation**)** हिलबर्ट वक्ररेषा

द्वितीय अंदाजानुसार हिलबर्ट वक्ररेषा काढण्यासाठी प्रत्येक चतुर्थांशाचे **(**Quadrants) विभाजन करून आणि पुढील प्रमुख चौकोनाकडे जाण्यापूर्वी त्यांची केंद्रे जोडून वक्ररेषा अंदाजे काढता येतात.आकृती क्र. .५.४ मध्ये द्वितीय अंदाजानुसार हिलबर्ट वक्ररेषा दिलेली आहे.



आकृती क्र .५.४ द्वितीय अंदाजानुसार (Second Approximation**)** हिलबर्ट वक्ररेषा

तृतीय अंदाजानुसार हिलबर्ट वक्ररेषा काढण्यासाठी पुन्हा प्रत्येक चतुर्थांशाचे **(**Quadrants) विभाजन करून आणि पुढील प्रमुख चौकोनाकडे जाण्यापूर्वी त्यांची केंद्रे जोडून वक्ररेषा अंदाजे काढता येतात.आकृती क्र. .५.५ मध्ये तृतीय अंदाजानुसार हिलबर्ट वक्ररेषा दिलेली आहे.



आकृती क्र.५. ५ तृतीय अंदाजानुसार (Third Approximation) हिलबर्ट वक्ररेषा

वरील आकृत्यांमधून आपण हिलबर्ट वक्ररेषा बद्दल खालील मुद्दे सहज लक्षात घेऊ शकतो:

1. जर आपण हिल्बर्टच्या वळणाचा अंदाज असीम विस्तारित केला**,** तर वक्र लहान चतुर्भुज भरते परंतु स्वतःला कधीही ओलांडत नाही.

2. वक्र स्क्रेअरमधील प्रत्येक बिंदूच्या अनियंत्रितपणे जवळ आहे.

3. वक्र ग्रिडवरील एका बिंदूमधून जातो**,** जो प्रत्येक उपविभागाप्रमाणे दुप्पट होतो.

4. उपविभागांना मर्यादा नाही आणि म्हणून वक्र लांबी अनंत आहे.

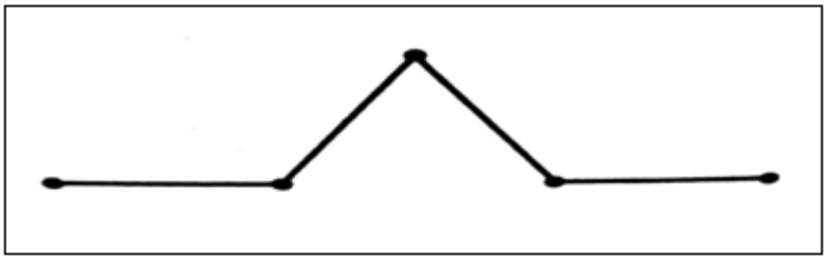
5. वक्र प्रत्येक उपविभागाच्या लांबीसह चारच्या घटकाने वाढते.

6. प्रत्येक उपविभागात स्केल **2** ने बदलते परंतु लांबी **4** ने बदलते म्हणून हिल्बर्टच्या वक्र टोपोलॉजिकल परिमाण एक आहे परंतु भग्न परिमाण **2** आहे**.**

५.२.२ कोच वक्ररेषा(Koch Curve)

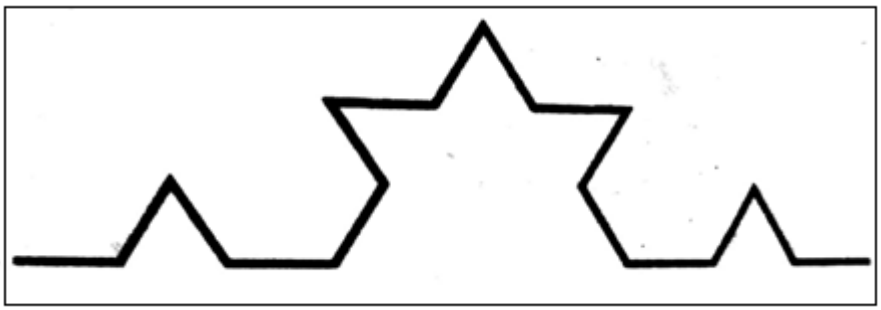
कोच वक्ररेषा तयार करण्यासाठी अंदाज पध्दतीचा वापर केला जातो . यामध्ये आपण सरळ रेषा चार समान भागामध्ये scaling फॅक्टर १/३ घेणून विभाजन अशाप्रकारे करतो की ,मध्यभागात दोन रेषाखंड असे काही समायोजित (adjusted) केले जातात ,ज्यामुळे दोन्ही रेषा समभुज त्रिकोनाच्या बाजू वाटतात . आकृती क्र ५. ४ पहा याच पध्दतीला प्रथम अंदाज (First Approximation) ने कोच वक्ररेषा निर्मिती करणे असे म्हणतात.

.



आकृती क्र.५.६ पहिल्या अंदाजानुसार कोच वक्ररेषा

अशाप्रकारे द्वितीय अंदाजाद्वारे (Second Approximation) कोच वक्ररेषा काढण्यासाठी पहिल्या अंदाजामधील प्रक्रियेची पुनरावृत्ती आपल्याला प्रत्येक चार रेषाखंडासाठी करावी लागेल . हि प्रक्रिया केल्यानंतर कोच वक्ररेषा आकृती क्र . ५.७ प्रमाणे तयार झालेली दिसेल.



आकृती क्र.५.७ दुसऱ्या अंदाजानुसार कोच वक्ररेषा

वरील आकृती मधील वक्ररेषा अधिक वळणावळणाची (Wiggles) आहे तसेच या वक्ररेषेची लांबी खऱ्या असणाऱ्या लांबीच्या १६/९ पट इतकी असते .

दिलेलेयाआकृत्यांमधून आपण कोच वक्ररेषे बद्दल खालील मुद्दे सहज लक्षात घेऊ शकतो:

१. प्रत्येक पुनरावृत्तीमुळे वक्ररेषेची लांबी **4/3** घटकाने वाढते.

२. वक्ररेषेची लांबी अनंत आहे.

३**.** हिल्बर्टच्या वक्ररेषेची निर्मिती करताना आपण चौरस भागाचे विभाजन करून मध्यवर्ती बिंदू एकमेकांना जोडतो म्हणजेच क्षेत्र **(**area**)** भरतो(fill) . अशी क्रिया कोच वक्ररेषे मध्ये होत नाही.

४**.** कोच वक्ररेषा त्याच्या मूळ आकारापासून फारसे विचलित होत नाही**.**

५**.** जर आपण वक्ररेषेचे प्रमाण **3** ने कमी केले**,** तर तयार झालेली वक्ररेषा जे अगदी मूळ वक्ररेषेसारखी दिसते परंतु आपल्याला चार अशा वक्रेषा एकत्र केल्या पाहिजे जेणेकरून तो आकार मूळ वक्ररेषेसारखा दिसेल.

त्यामुळे,आपण खालील प्रमाणे समीकरण लिहितो

$$4 = 3D$$

वरील समीकरण जर d साठी सोडवल तर d साठी खालील मिळेल

$D = \log_3 4 = \log 4 / \log 3 = 1.2618$

म्हणून कोच वक्ररेषेचे टोपोलॉजिकल डायमेंशन १ आहे पण फ्रॅक्टल डायमेंशन १.२६१८ आहे.

वरील विवेचनावरून आपण असे म्हणू शकतो की बिंदूचे वक्र आणि पृष्ठभाग जे टोपोलॉजिकल परिमाणापेक्षा जास्त परिमाण देतात त्यांना फ्रॅक्टल **(**fractals) असे म्हणतात. हिलबर्ट वक्ररेषा आणि कोच वक्ररेषा या फ्रॅक्टल्स आहेत. कारण त्यांची फ्रॅक्टल आकारमान **(**fractal dimensions)(क्रमाक्रमाने २ आणि १.२६१८) ही त्यांच्या टोपोलॉजिकल आकारमानापेक्षा जास्त आहेत जे की 1 आहे.

५.२.३ बी-Spline वक्ररेषा(B-spline Curve)

आपण पाहिले आहे की, वक्र आणि बहुभुज यांच्यातील संबंध प्रस्थापित करण्यासाठी काही इंटरपोलेशन(interpolation) किंवा अंदाजे योजनेवर अवलंबून असलेल्या परिभाषित बहुभुजाच्या शिरोबिंदूंचा वापर करून वक्र तयार केला जातो. ही योजना बेस फंक्शनच्या (basis function)निवडीद्वारे प्रदान केली जाते. बर्नस्टीन(Bernstein) बेस फंक्शनद्वारे(basis function) निर्मित झालेल्या बेझियर वक्रेषमध्ये मर्यादित लवचिकता (flexibility)असते.

प्रथम निर्दिष्ट (specified) बहुभुज (polygon) शिरोबिंदूंची संख्या परिणामी बहुपदीचा(polynomial) क्रम निश्चित करते ज्याचा उपयोग वक्ररेषा परिभाषित(define) करण्यासाठी होतो . उदाहरणार्थ, चार शिरोबिंदू असलेल्या बहुभुजामुळे घन बहुपदी असलेली वक्ररेषा तयार होते. वक्राचा अंश कमी करण्याचा एकमेव मार्ग म्हणजे शिरोबिंदूंची संख्या कमी करणे आणि याउलट वक्राची पदवी(degree) वाढवण्याचा एकमेव मार्ग म्हणजे शिरोबिंदूंची संख्या वाढवणे. दुसरी मर्यादित वैशिष्ट्ये म्हणजे संपूर्ण वक्रवरील सर्व पॅरामीटर मूल्यांसाठी ब्लेंडिंग (blending) फंक्शनचे मूल्य शून्य असते. एका शिरोबिंदूमधील या बदलामुळे, संपूर्ण वक्र बदलते आणि यामुळे वक्रामध्ये स्थानिक बदल करण्याची क्षमता नाहीशी होते.

आणखी एक बेस फंक्शन आहे, ज्याला बी-स्प्लाइन बेसिस म्हणतात, ज्यामध्ये बर्नस्टाईन(Bernstein) बेस विशेष केस म्हणून समाविष्ट आहे. बी-स्प्लाइनचा आधार नॉनग्लोबल आहे. हे नॉनग्लोबल आहे कारण प्रत्येक शिरोबिंदू B एका अद्वितीय आधार कार्याशी संबंधित आहे. अशाप्रकारे, प्रत्येक शिरोबिंदू वक्रच्या आकारावर केवळ पॅरामीटर मूल्यांच्या श्रेणीवर परिणाम करतो जेथे त्याचे संबंधित आधार कार्य शून्य असते. बी-स्प्लाइन आधार देखील बेस फंक्शनच्या क्रमास अनुमती देतो आणि म्हणून परिणामी वक्रची डिग्री शिरोबिंदूंच्या संख्येवर स्वतंत्र असते. परिभाषित बहुभुजाच्या शिरोबिंदूंची संख्या न बदलता परिणामी वक्रची डिग्री बदलणे शक्य आहे.

आपण बी-Spline वक्ररेषा काढण्यासाठी खालील गणितीय समीकरण पाहूया.

If P(u) be the position vectors along the curve as a function of parameters u, a B-spline curve is given by

$$P(u) = \sum_{i=1}^{n+1} B_i N_{i,k}(u) \quad u_{min} \le u < u_{max}, \quad 2 \le k \le n+1$$

Where the B_i are the position vectors od n+1 defining polygon vertices and the $N_{i,k}$ are the normalized B-spline basis functions.

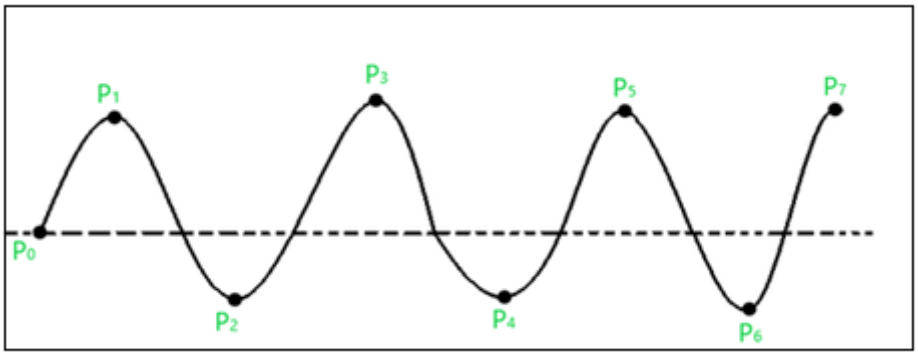
For i^{th} normalized B-spline basis function of order k, the basis function $N_{i,k}$ (u) are defined as

$$N_{i,1}(u) = \begin{cases} 1 & \text{if } x_i \le u < x_{i+1} \\ 0 & \text{Otherwise} \end{cases}$$

and

$$N_{i,k}(u) = \frac{(u - x_1) N_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u) N_{i+1,k-1}(u)}{x_{i+k} - x_{i+1}}$$

बी-स्प्लाइन वक्ररेषा ही संकल्पना बेझियर वक्रेषे द्वारे असणा-या गैरसोयींचे निराकरण करण्यासाठी आली आहे, कारण आपल्या सर्वांना माहित आहे की दोन्ही वक्र निसर्गात पॅरामेट्रिक आहेत. बेझियर वक्रमध्ये आपल्याला एक समस्या येते, जेव्हा आपण कोणतेही नियंत्रण बिंदूचे संबंधित स्थान बदलतो तेव्हा संपूर्ण वक्ररेषेचा आकार बदलतो. परंतु येथे बी-स्प्लाइन वक्ररेषेमध्ये, वक्र-आकाराचा एक विशिष्ट विभाग बदलतो किंवा नियंत्रण बिंदूंच्या संबंधित स्थानाच्या बदलामुळे प्रभावित होतो.आकृती क्र.५.८ मध्ये आपण बी-स्प्लाइन वक्ररेषा पाहू शकतो.



आकृती क्र. ५.८ बी-Spline वक्ररेषा

बी-Spline वक्ररेषेचे गुणधर्म

१. कोणत्याही पॅरामीटर मूल्यासाठी बी-स्प्लाइन बेस फंक्शन्सची बेरीज 1 आहे.

२. प्रत्येक आधारभूत कार्य सर्व पॅरामीटर मूल्यांसाठी सकारात्मक किंवा शून्य आहे.

३. प्रत्येक बेस फंक्शनमध्ये k=1 वगळता तंतोतंत एक कमाल मूल्य असते.

४. वक्राचा कमाल क्रम बहुभुज परिभाषित करणाऱ्या शिरोबिंदूंच्या समान आहे.

५. बहुभुज परिभाषित करणाऱ्या शिरोबिंदूंची संख्या आणि बी-स्प्लाइन बहुपदीची डिग्री स्वतंत्र आहेत.

६. वक्र पृष्ठभागावरील स्थानिक नियंत्रणास बी-स्प्लाइनद्वारे अनुमती आहे, कारण प्रत्येक शिरोबिंदू वक्रच्या आकारावर परिणाम करतो आणि जेथे संबंधित आधार कार्य शून्य असते.

७. भिन्नता कमी करणारी property या वक्ररेषे द्वारे प्रदर्शित केली जाते.

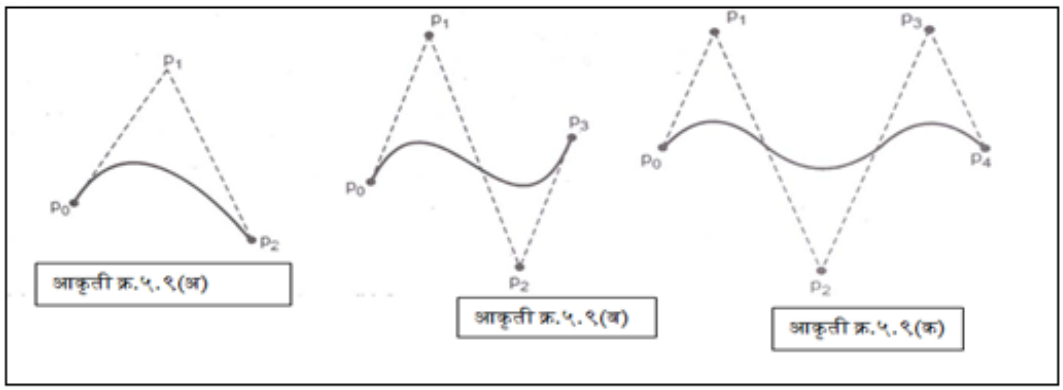
८.परिभाषित बहुभुजाचा आकार खालीलप्रमाणे आहे.

९.बहुभुज परिभाषित करण्याच्या शिरोबिंदूंना लागू करून, वक्र वर एक affine परिवर्तन लागू केले जाते

५.२.४ बेझियर वक्ररेषा(Bezier Curve)

वक्ररेषा काढण्यासाठी बेझियर वक्ररेषा ही आणखी एक पद्धती आहे. बेझियर वक्ररेषा परिभाषित (define) करण्यासाठी बहुभुजाद्वारे (polygon) निर्धारन (determines) केले जाते. बेझियर वक्ररेषेमध्ये अनेक गुणधर्म आहेत जे त्यांना वक्र आणि पृष्ठभागाच्या डिझाइनसाठी अत्यंत उपयुक्त आणि सोयीस्कर बनवतात. त्यांची अंमलबजावणी करणे देखील सोपे आहे. म्हणून बेझियर वक्ररेष विविध CAD प्रणालींमध्ये आणि सामान्य ग्राफिक पॅकेजेसमध्ये मोठ्या प्रमाणावर उपलब्ध आहेत. या विभागात आपण क्यूबिक बेझियर(Cubic Bezier) वक्र बद्दल चर्चा करू. क्यूबिक बेझियर वक्र निवडण्याचे कारण म्हणजे ते वाजवी डिझाइन लवचिकता (reasonable design flexibility) प्रदान करतात आणि मोठ्या संख्येने गणना(Calculation) टाळतात.

सर्वसाधारणपणे, बेझियर वक्र भाग कितीही नियंत्रण बिंदूंवर बसविला जाऊ शकतो. तथापि, नियंत्रण बिंदूंची संख्या जसजशी वाढते तसतसे बेझियर बहुपदाची पदवी (polynomial degree) देखील वाढते. कारण बेझियर वक्र मध्ये बहुपदीची डिग्री वापरलेल्या नियंत्रण बिंदूंच्या संख्येपेक्षा एक कमी असते. उदाहरणार्थ, तीन नियंत्रण बिंदूपासून एक पॅराबोला (parabola) तयार होतो, चार बिंदू क्यूबिक वक्ररेषा निर्माण करतात इत्यादी. हे आकृती क्र. ५.९ मध्ये स्पष्ट केले आहे.आकृती क्र.५.९(अ) मध्ये बेझीयर वक्ररेषा ही तीन नियंत्रण बिंदूंनी बनवलेलीआहे. आकृती क्र.५.९(ब) मध्ये बेझीयर वक्ररेषा ही चार नियंत्रण बिंदूंनी बनवलेली आहे. आकृती क्र.५.९(क) मध्ये बेझीयर वक्ररेषा ही पाच नियंत्रण बिंदूंनी बनवलेली.



आकृती क्र.५.९ बेझियर वक्ररेषा (तीन,चारआणि पाच नियंत्रण बिंदू वापरून)

बेझियर वक्ररेषा काढण्यासाठी समीकरण (Bezier Curve Equation)

A Bezier curve is parametrically represented by

$$P(t) = \sum_{i=0}^{n} B_i \, J_{n,i}(t)$$

Here,

- t is any parameter where 0 <= t <= 1
- P(t) = Any point lying on the bezier curve
- B_i = i^{th} control point of the bezier curve
- n = degree of the curve
- $J_{n,i}(t)$ = Blending function = $C(n,i)\, t^i\, (1-t)^{n-i}$ where $C(n,i) = n! / i!(n-i)!$

बेझियर वक्ररेषेचे गुणधर्म (Properties of Bezier Curve)

१ ते नेहमी पहिल्या आणि शेवटच्या नियंत्रण बिंदूंमधून जातात.

२. ते त्यांच्या परिभाषित नियंत्रण बिंदूंच्या बहिर्वक्र हुलमध्ये समाविष्ट आहेत.

३. वक्र विभाग परिभाषित करणाऱ्या बहुपदीची डिग्री परिभाषित करणाऱ्या बहुभुज बिंदूच्या संख्येपेक्षा एक कमी आहे. म्हणून, 4 नियंत्रण बिंदूंसाठी, बहुपदीची पदवी 3 आहे, म्हणजे घन बहुपदी.

४. एक बेझियर वक्र सामान्यतः परिभाषित बहुभुजाच्या आकाराचे अनुसरण करते

५. शेवटच्या बिंदूंवरील स्पर्शिका वेक्टरची दिशा पहिल्या आणि शेवटच्या खंडांद्वारे निर्धारित केलेल्या वेक्टर सारखीच असते.

६. बेझियर वक्र जागतिक(ग्लोबल) नियंत्रण प्रदर्शित करतात म्हणजे नियंत्रण बिंदू हलवल्याने संपूर्ण वक्र आकार बदलतो.

तक्ता क्र. ५.१ बी-spline वक्ररेषा आणि बेझियर वक्ररेषा याच्यांतील फरक खालील प्रमाणे.

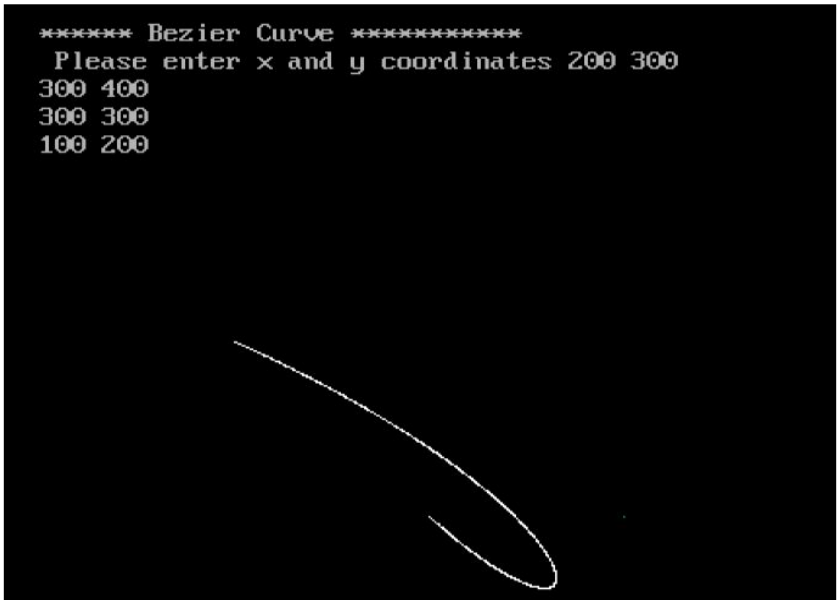
अ.क्र.	बी-spline वक्ररेषा	बेझियर वक्ररेषा
१	बी-स्प्लाइन वक्ररेषा मर्यादित लवचिकता(flexibility) असलेल्या बर्नस्टाईन बेस फंक्शनद्वारे specify केले जातात.	बेझियर वक्र सीमा परिस्थितीसह, वैशिष्ट्यपूर्ण मॅट्रिक्ससह किंवा ब्लेंडिंग फंक्शनसह निर्दिष्ट केले जाऊ शकतात.
२	हे वक्र ओपन युनिफॉर्म बेस फंक्शनच्या वापराचा परिणाम आहेत.	वक्र साधारणपणे परिभाषित बहुभुजाच्या आकाराचे अनुसरण करते.
३	बी-स्प्लाइन बेस फंक्शनच्या क्रमाला अनुमती देते आणि म्हणून परिणामी वक्राची डिग्री शिरोबिंदूंच्या संख्येपासून स्वतंत्र असते.	वक्र विभाग परिभाषित करणाऱ्या बहुपदीची डिग्री परिभाषित करणाऱ्या बहुभुज बिंदूच्या संख्येपेक्षा एक कमी आहे.
४	बी-स्प्लाइन, वक्र पृष्ठभागावर स्थानिक नियंत्रण असते आणि वक्रचा आकार प्रत्येक शिरोबिंदूवर प्रभावित होतो	हे संबंधित फील्डमध्ये वापरले जाणारे पॅरामेट्रिक वक्र आहे.

चार नियंत्रण बिंदू(control ponits) वापरून बेझियर वक्ररेषा काढण्यासाठी C प्रोग्राम खालील प्रमाणे

```
#include<graphics.h>
#include<math.h>
#include<conio.h>
#include<stdio.h>
void main()
{
int x[4],y[4],i;
double put_x, put_y,t;
int gr=DETECT,gm;
initgraph (&gr,&gm,"C:\\TURBOC3\\BGI");
printf("\n****** Bezier Curve **********");
printf("\n Please enter x and y coordinates ");
```

```
for(i=0;i<4;i++)

{

scanf("%d%d",&x[i],&y[i]);

putpixel(x[i],y[i],3);              // Control Points

}

for(t=0.0;t<=1.0;t=t+0.001)          // t always lies between 0 and 1

{

put_x = pow(1-t,3)*x[0] + 3*t*pow(1-t,2)*x[1] + 3*t*t*(1-t)*x[2] + pow(t,3)*x[3]; // Formula to draw curve

put_y =  pow(1-t,3)*y[0] + 3*t*pow(1-t,2)*y[1] + 3*t*t*(1-t)*y[2] + pow(t,3)*y[3];

putpixel(put_x,put_y, WHITE);          // putting pixel

}

getch();

closegraph();

}
```

बेझियर वक्ररेषे च्या C प्रोग्राम चे **Output**



HEAD OFFICE

Secretary,
Maharashtra State Board of Technical Education
49, Kherwadi, Bandra (East), Mumbai - 400 051
Maharashtra (INDIA)
Tel: (022)26471255 (5 -lines)
Fax: 022 - 26473980
Email: -secretary@msbte.com
Web -www.msbte.org.in

REGIONAL OFFICES:

MUMBAI

Deputy Secretary (T),
Mumbai Sub-region,
2nd Floor, Govt. Polytechnic Building,
49, Kherwadi, Bandra (East)
Mumbai - 400 051
Phone: 022-26473253 / 54
Email: rbtemumbai@msbte.com

PUNE

Deputy Secretary (T),
M.S. Board of Technical Education,
Regional Office,
412-E, Bahirat Patil Chowk,
Shivaji Nagar, Pune
Phone: 020-25656994 / 25660319
Fax: 020-25656994
Email: rbtepn@msbte.com

NAGPUR

Deputy Secretary (T),
M.S. Board of Technical Education
Regional Office,
Mangalwari Bazar, Sadar, Nagpur - 440 001
Phone: 0712-2564836 / 2562223
Fax: 0712-2560350
Email: rbteng@msbte.com

AURANGABAD

Deputy Secretary (T),
M.S. Board of Technical Education,
Regional Office,
Osmanpura, Aurangabad -431 001.
Phone: 0240-2334025 / 2331273
Fax: 0240-2349669
Email: rbteau@msbte.com